Abalone Classification Model Applying Decision Trees and Random Forest

DMML Assessment 3 Report

Ardeshir Bozorgmehr
School of Mathematics and Statistics
University of New South Wales
Sydney, Australia
Ardeshir.Bozorgmehr@student.unsw.edu.au

***Abstract* — This report investigates the classification of abalone age with eight other features including age, diameter, etc. It is difficult to estimate the age of abalone and in this article, we seek to classify them into four classes. The applied methodologies are primarily based upon concepts in machine learning i.e., decision trees and ensemble learning, e.g., random forest. The underlying algorithm is CART with the GINI method. Model accuracies are tested, and results suggest over 60 percent accuracy in the test set is achievable. The results shared are saved using software packages in Python and are reproducible. The data source used is publicly accessible via the repository of the University of California at Irvine. The research suggests that using random forests would lead to a better model in terms of higher classification accuracies as well as lower generalization errors.**

Keywords* — *Decision Tree, Random Forest, Machine Learning, Cost Complexity Pruning, Cross-Validation, Classification

I. Introduction

In the 1990s, techniques to classify tasks were introduced to help decision-makers view the knowledge in the form of decision trees [2]. With the growing computational power and versatility of ensemble methods, problems in a wide spectrum of domains were investigated and modelled [3]. The advent of random forests for use in classification problems addressed the issues of generalization error by improving the strength of individual trees [4].

At the same time, some complexities remain as explaining the outcomes of ensemble methods and random forests in many cases remains a challenge [5].

In this report, I sought to apply decision trees and random forests to a dataset to classify the age of abalone into four categories using eight features. The abalone dataset was received in December 1995 and was owned by the Marine Resources Division and Research Laboratories – Taroona at the Department of Primary Fisheries in Hobart, Tasmania, Australia.

The motivation and goal were to develop and identify the most efficient decision tree and random forest models with minimization of generalization error (trade-off between the bias and variance) and test their accuracies under varied inputs conditions, hyperparameter tuning, cost-complexity pruning, etc. to best perform on the abalone allocation into quartet pre-defined classes.

Python and its related libraries e.g., Pandas, NumPy, Sci-Kit, Seaborn, and Matplotlib were used to code and run the models.

II. Methodology

Data: There are four pre-defined classes on Rings feature, in order:

- Class 1: 0 - 7 years,
- Class 2: 8- 10 years
- Class 3: 11 - 15 years
- Class 4: Greater than 15 years

This dataset has 4177 samples with 9 columns out of which the last one, Rings, is the dependent variable, and should its respective value be summed by 1.5, one would obtain the age in years. The first task was to assign numbers to males, females, and infants and then separate a 60-40% train test split to first train the model using 60% of the dataset and then after, for testing purposes, use the remaining 40%. Note that to keep the generalization error in check downstream, the usage of a validation set is required [5].

Decision trees are notoriously susceptible to overfitting the training data [1]. Given the properties of using one-hot encoding in accuracy improvement [6], I decided to use one-hot encoding for the categorical feature of “Sex” and turn it into three columns representing Infants, Females, and Males.

Then I constructed a decision tree both in a graphically presentable way and unconstrained model. Given the relatively low accuracies of the resulting model, I then applied pruning methods and k-fold cross-validation to find the best cost complexity pruning coefficient (namely CCP Alpha) to tune the tree performance.

The other method used is bagging trees via Random Forest. I examined a range of hyperparameters (e.g., number of trees, CCP Alpha) and further improve the model using the properties of ensemble methods. At the same time, in doing cross-examinations, I also modeled 1000 models to execute a thorough one-dimensional grid search since the value showed empirically capable in the estimation of test error [7].

As was mentioned, to replicate the results, researchers advised installing Python on their devices plus some commonly used libraries related to the field of data science and machine learning including Pandas, NumPy, Sci-Kit as well as those related to the graphical presentation of models and function such as Seaborn and Matplotlib. Python is free software, and all the above libraries are also publicly accessible for download and use.

III. Problem Definition

A. Report the Tree Visualisation (show your tree and also translate a few selected nodes and leaves into IF and Then rules)

The first task was to build a decision tree and demonstrate its performance. To do this, I used a validation set approach to keep generalization errors in check. The tree depth was limited to 5 so the image remains readable. Some results are presented via If/Then statements to show the test report of underlying rules in the decision tree.

B. Do an investigation about improving performance

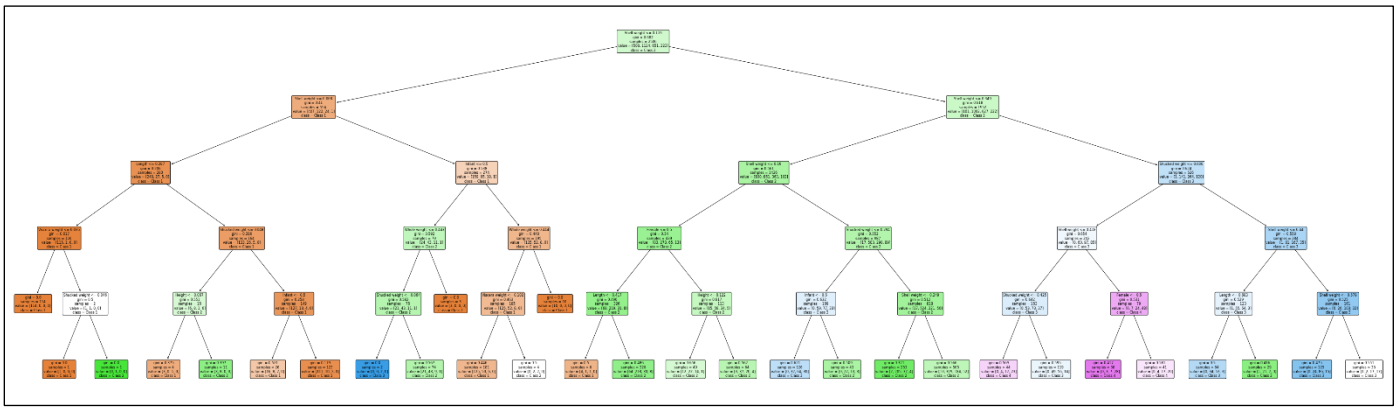

Figure 1 - Decision Tree

further by pruning the tree

In part 2, pruning as a regularization method is introduced to find the best cost complexity pruning Alpha (hyperparameter) for the cost-complexity pruning. At this stage, cross-validation is used to investigate the accuracy certainty (mean and standard deviation of accuracy) in the results to check how reliable the model output is in face of different training and test (validation) sets.

C. Apply Bagging of Trees via Random Forests and show performance as the number of trees in the ensembles increases. Carry out 10 experiments in Parts 1 and 3 and show performance with mean and confidence intervals.

The researcher would first use the original train-test split to fit about 700 models with a different number of trees in the forest and plot the respective model accuracies and RMSE results. Then, develop and fit the data using the optimal number of trees to review the resulting confusion matrix and discuss the findings.

At the same time, given the variation in accuracies using 5-fold cross-validation, I sought to find the best complexity parameter for pruning in random forest.

Finally, 10 different train-test splits are created and both decision tree and random forest models are fit to investigate the mean and variance (with/without best CCP Alpha).

IV. Results

Using the Abalone dataset from the University of California Irvine repository, I first show the processed data into four classes and plot a histogram [Fig. 1]:

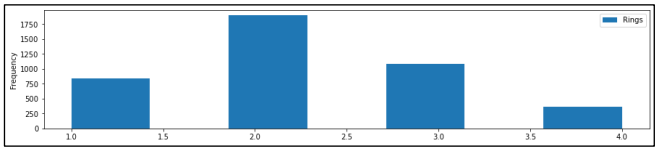

Figure 2 - Histogram

It can be seen the majority of abalone belong to class 2 with ring-age between 8 to 10 years with only a minority reaching 15 and over.

At this stage, without normalizing the data, I constructed a model (with max depth = 5) and visualized the decision tree.

It can be read using If, Then ruling that:

|--- *Shell weight <= 0.11*
| |--- *Diameter <= 0.25*
| | |--- *Shell weight <= 0.04*
| | | |--- *Shell weight <= 0.03*
| | | | |--- *Male <= 0.50*
| | | | | |--- *weights: [86.00, 0.00, 0.00, 0.00] class: 1*
| | | | |--- *Male > 0.50*
| | | | | |--- *weights: [16.00, 1.00, 0.00, 0.00] class: 1*

IF(*Shell weight* is smaller or equal to 0.03, *AND, Diameter* is smaller or equal to 0.25) THEN the abalone belongs to Class 1 (irrespective of Male column). Please note that given the introduction of 3 new columns for *Sex* categories using one-hot-encoding, the interpretation of the resulting decision tree becomes slightly more difficult, however, the benefits outweigh the drawbacks in making the model more mathematically as binary digits in these columns would convey the message if the property (i.e. being a male or female or infant) applies to the respective abalone or not.

To better understand the performance of the model, I removed the constraint of depth equal to 5, re-trained the model, fitted the test dataset, and finally print the confusion matrix [Fig. 3]; it can be seen that the model score is marginally more than 52.78% with the details being:

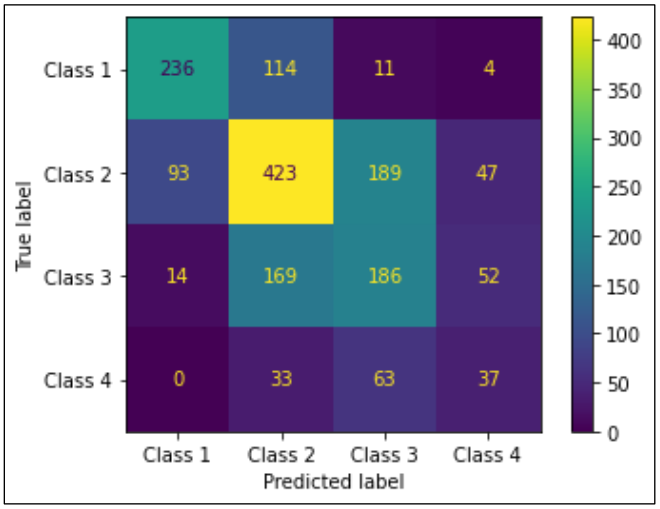

Figure 3- Confusion Matrix for Non-Normalized Data Using Decision Tree

The result depicts [Fig.3] a rather satisfactory precision and recall for the first two classes yet, poor performance in the latter two classes.

Now, given the poor performance of the decision tree, we would like to review the accuracy of the model with cost-complexity-pruning in a range of alpha. Hence, I modelled a range of alphas except for the largest one (as it would prune all branches and leave only the leaf) concerning the accuracy of the test and train sets. The result is:

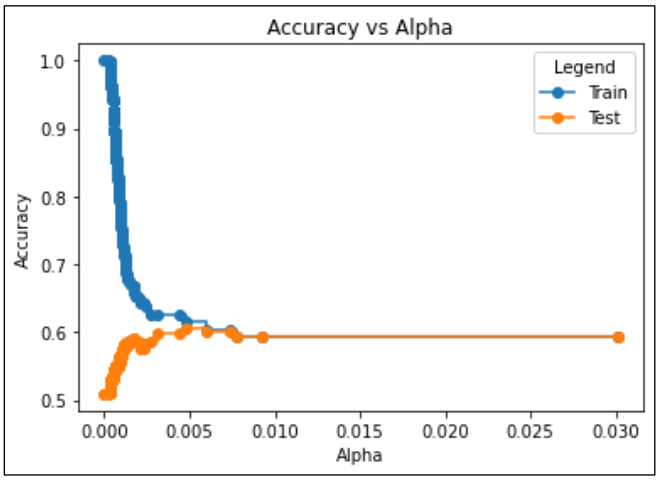

Figure 4 - Accuracy vs Alpha (Decision Tree Pruning)

While the alpha is zero, determining how much pruning to happen in the model, the train set overfits the data, and the test set consequently suffers from high variance (test model accuracy is 50%). However, as the pruning amount slightly increases, we see a substantial decrease in overfitting while test accuracies also improved, suggesting we are on the right track to improving the model accuracy. However, again as alpha grows too much, excessive pruning causes the tree to be further pruned and lose bias and hence accuracy.

At this stage, we note that we could have broken down the train test split in a different order. As such, all the results above belong to one specific train test split that if changed, the model results could be different! One approach is to use k-fold cross-validation to see if the model with pruning can still deliver stable accuracy results on a broad range of data splits like that above! As a result, we run a 5-fold cross-validation on the dataset that trains and tests the model to see the impact:

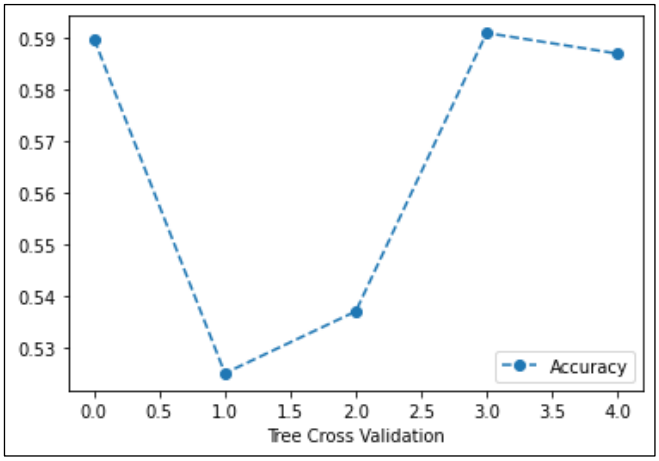

Figure 5 - Accuracy vs Tree 5-Fold Cross-Validation

There is rather a notable fluctuation in test accuracy results! As such the best alpha that was found may not be necessarily the best pruning coefficient for a different test set!

The solution is to execute multiple runs (cross-validating 5 folds) and see the mean and standard deviation and as per this average accuracy result, we select the best pruning alpha. So, we have:

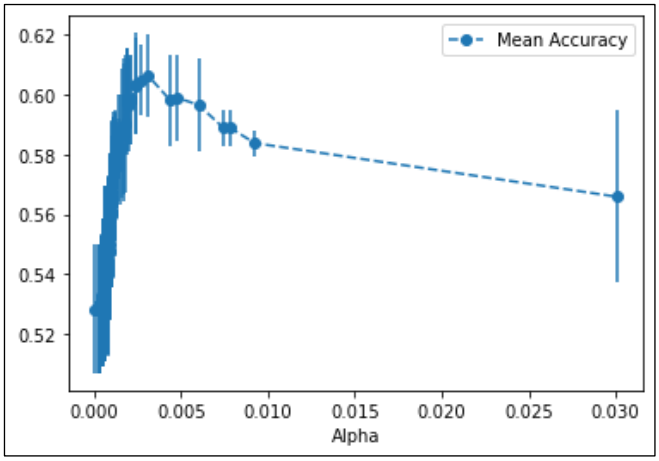

Figure 6 - Accuracy Average and Std. Deviation vs 5-Fold Cross Validation Alpha Values

Table 1 - Best Alpha for Decision Tree Information

Best Alpha	Resulting Accuracy	Index
0.003082909075209306	0.60654308912056	361

I constructed a model with this alpha and expected to see better more stable accuracies over more train-test splits. Consequently, I run a model using the best alpha and the accuracy went up to 59.84% the confusion matrix is as follows:

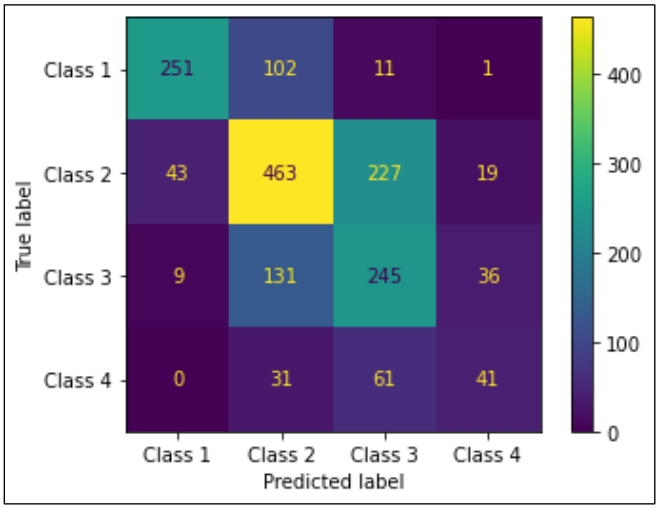

*Figure 7- Confusion Matrix for Best **Decision Tree Model** using Optimal Alpha [Cost-Complexity-Pruning] Value*

This model demonstrates approximately 7% improvement in the accuracy of a single decision tree when the suitable cost complexity pruning coefficient is used. The detailed performance information is as follows:

Table 2- Best Decision Tree Model Table

Class	Precision	Recall	F1-score	Support
1	0.83	0.69	0.75	365
2	0.64	0.62	0.63	752
3	0.45	0.58	0.51	421
4	0.42	0.31	0.36	133
Macro Average Accuracy	0.58	0.55	0.56	1671
Weighted Average Accuracy	0.61	0.60	0.60	1671

Now that we created a satisfactory decision tree, I shifted the model structure to ensemble learning and use bagging of trees and Random Forest to see if a better model could be built. Random Forest models use bagging which is great for times that perturbation in the learning set can cause changes in predictors constructed and using bagging improves accuracy [12]. The first step is to investigate the impact of the number of trees on the accuracy of the resulting Random Forest. Hence, I created 1000 Random Forest models with 1 to 1001 bagged trees and plotted the resulting train, test root-mean-square errors (RMSE) as well as the train and test accuracies. The result is as follows:

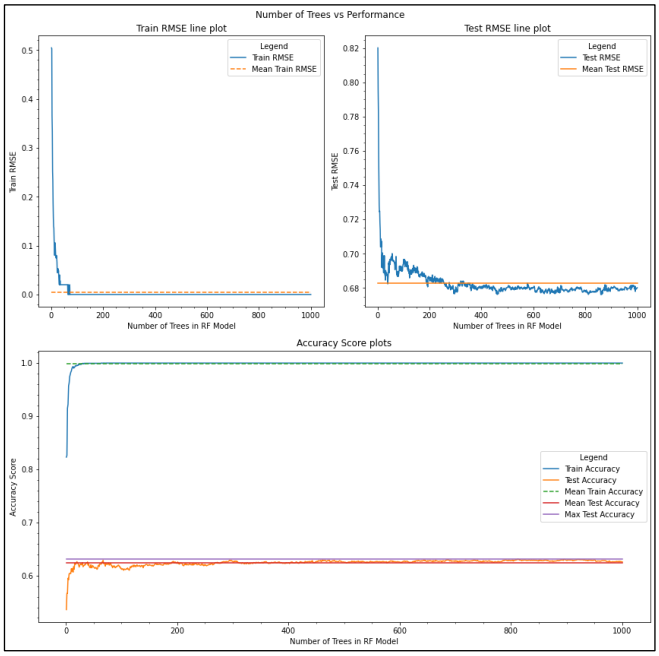

Figure 8- Number of Trees vs Train/Test RMSE and Accuracy Score

1000 models with multiplications of 1 tree to 1001 are plotted, with a maximum number of trees in the final model being 1001. In the beginning, the test accuracies are mostly below the red Mean Test Accuracy. However, as the number of trees in Random Forest models increases, the test accuracy improves. Note that the improvement has some fluctuation yet as the overall trend is upwards. The best accuracy is obtained when the random forest model has 812 bagged trees. When the number of trees reaches about 65, the train test RMSE drops to zero meaning the resulting random forests are fully explaining the train sets. However, the test set RMSE reduced to about 0.685 and then after, very slowly reduces with some variations. On Accuracy Score, I have the model's best number of trees being 67 with an accuracy of 63.076%. As such, with that number of trees, I built the model and further dived into its confusion matrix and details:

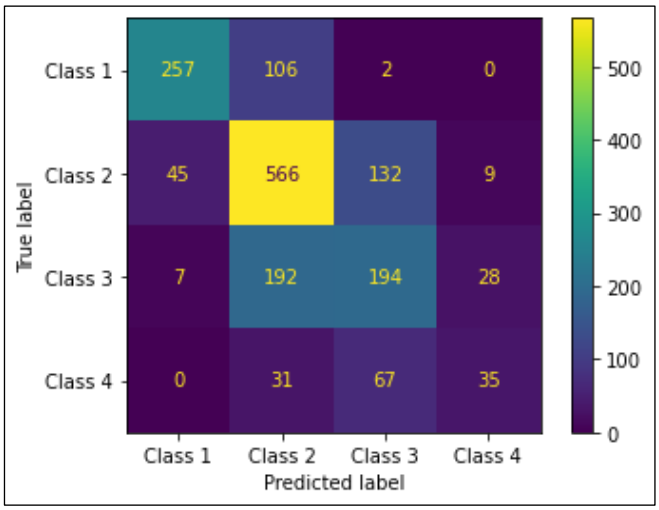

Figure 9- Confusion Matrix of Best Random Forest Without Pruning out of 1000 Models

Table 3- Not Pruned Best Random Forest Information

Class	Precision	Recall	F1-score	Support
1	0.84	0.70	0.76	365
2	0.64	0.79	0.70	752
3	0.52	0.46	0.49	421
4	0.49	0.22	0.30	133
Macro Average Accuracy	0.62	0.54	0.56	1671
Weighted Average Accuracy	0.64	0.64	0.63	1671

From Table 3, an immediate improvement in the accuracy score of the Random Forest model vs best decision tree by approx. 3.2% is substantiated. Moreover, note that the precision of class 3 and especially class 4 improved while for class 1 and 2, numbers remained the same. As with the decision tree models, the question now is how reliable this result is assuming a different mixture of train-test split to be used and if pruning could be of some help here!

Now, I decided to investigate if setting max depth could improve the model and realized that if it is set at 9, the test accuracy would further improve and reach 63.794%

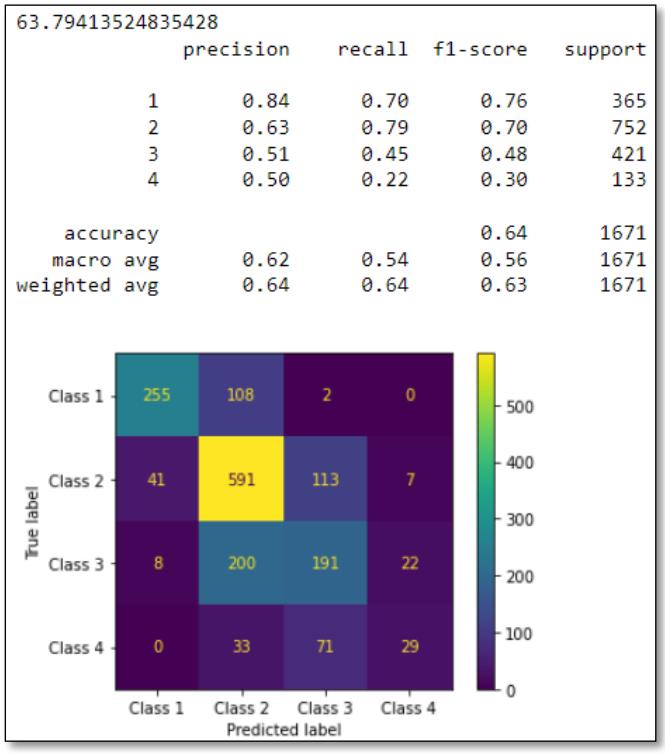

Figure 10 - Confusion Matrix after Introduction of Max Depth 9 (Model Accuracy at 63.8%)

At this stage, I defined a range for CCP Alphas to be investigated by the model as CCP (Cost-Complexity-Pruning) coefficient and then fed the alpha to the Random Forests with 812 trees and track the respective effect on accuracy.

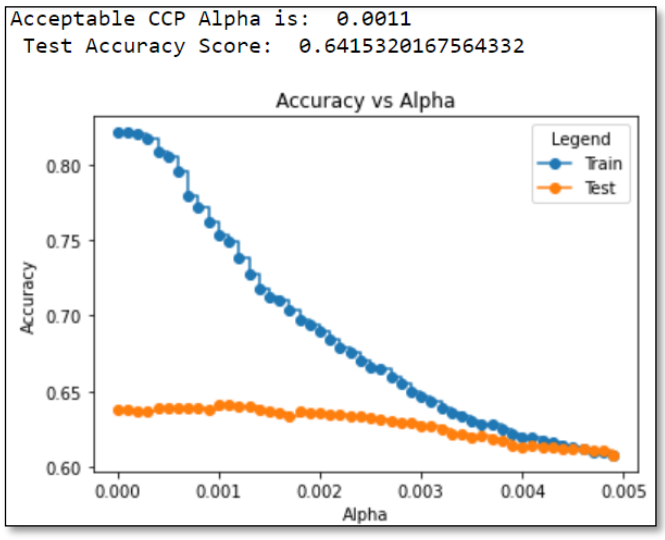

Figure 11 - Accuracy vs Cost Complexity Pruning in Random Forest model with Optimal Number of Trees

This semi-grid search of range 50 on alpha values starting at *0.000* to *0.005* shows that the application of cost complexity pruning to the model will cause rather a marginal improvement!

In comparison, the models using Random Forest have better stabilities compared with Decision Trees and CCP Alpha is not making as much difference as with the 7% that was delivered in the decision tree model after pruning. Here, using the right CCP Alpha would suggest an improvement in test accuracy by 0.5% to 64.15% test accuracies.

From the result above [Fig.11], I further narrowed down the hyperparameter tuning and investigate 20 more models with alpha increments of 0.00001 to identify any potential better model. The result came as:

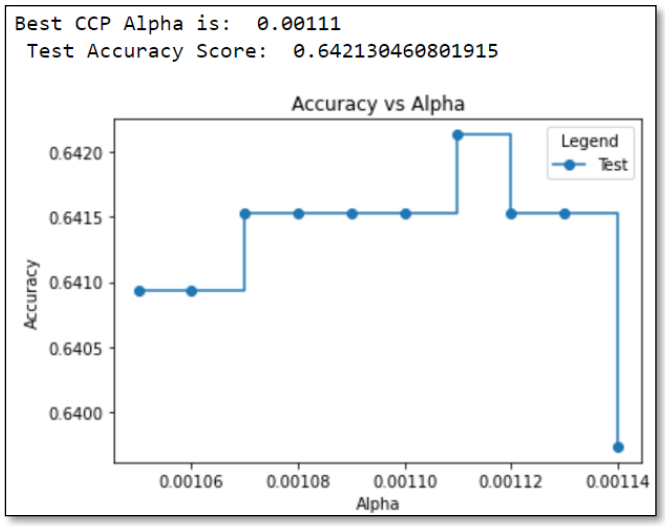

Figure 12 - Best CCP Alpha hyperparameter tuning

Using the above detail to build the best Random Forest model with 812 trees, max depth at 9, and CCP Alpha at 0.00111, one would get the below [Fig. 13] confusion matrix with an overall accuracy of **64.21**%:

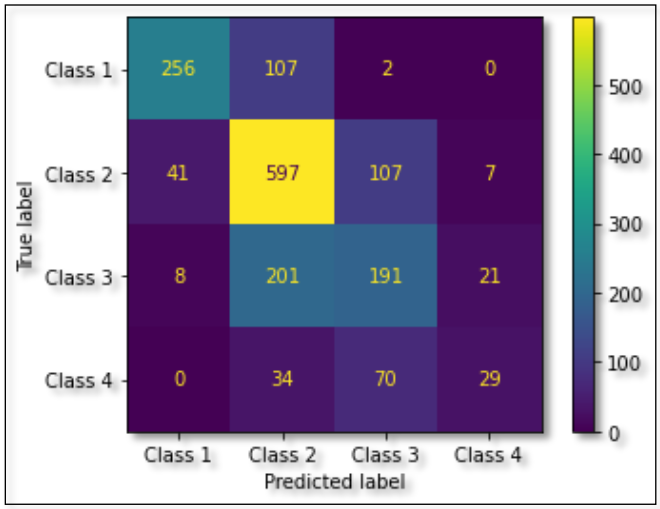

*Figure 13 - Confusion Matrix Using Best **Random Forest Model***

Table 4 - Pruned Random Forest Model Information:

Class	Precision	Recall	F1-score	Support
1	0.84	0.70	0.76	365
2	0.64	0.79	0.71	752
3	0.52	0.45	0.48	421
4	0.51	0.22	0.31	133
Macro Average Accuracy	0.63	0.54	0.56	1671
Weighted Average Accuracy	0.64	0.64	0.63	1671

It is worth mentioning that in this model [Table 4], the Precision of class 1 correctly being classified has reached 84% and the recall is 70%.

Finally, I also reviewed the models using 10 different train-test splits and calculate the average accuracy of the Decision Tree and Random Forest best models.

For the decision trees the result returned as:

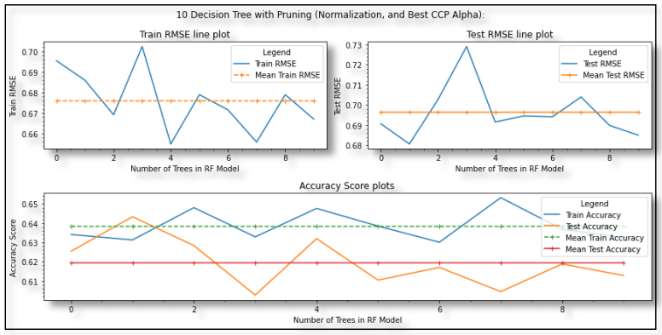

Figure 14 - 10 Decision Tree with Pruning (Normalization, and Best CCP Alpha)

Table 5 - 10 Decision Tree with Pruning

Mean Test Accuracy Score :	0.619509275882705
Standard Deviation Test Accuracy Score:	0.01217300503659233
Mean Train Accuracy Score:	0.6385075818036712
Standard Deviation Train Accuracy Score:	0.007775592871138949
Mean Test RMSE:	0.6961570903158254
Standard Deviation Test RMSE:	0.012838490867987645

And for random forests we have:

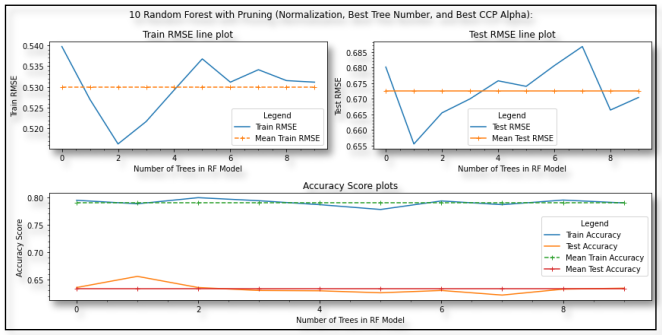

Figure 15 - 10 Random Forest with Pruning (Normalization, Best Tree Number, Max Depth, and Best CCP Alpha)

Table 6 - 10 Random Forest with Pruning

Mean Test Accuracy Score :	0.6387791741472173
Standard Deviation Test Accuracy Score:	0.003880207080178895
Mean Train Accuracy Score:	0.7472067039106145
Standard Deviation Train Accuracy Score:	0.006345274466035075
Mean Test RMSE:	0.6747606689576275
Standard Deviation Test RMSE:	0.007356210635677373

The result from running splitting 10 different train-test sets suggests that with 68% confidence, the Random Forest model test accuracy falls within the interval of (63.49% - 64.26%) test accuracy with mean accuracy being 63.88%. In the Decision Tree model, at 68% confidence, the model test accuracy would be in the range of (60.73% - 63.17%) averaging at 61.95%. Note that 95.44% confidence intervals are as follows:

Table 7 - 95.44% Test Confidence Intervals in 10 Train/Test Split Runs

Decision Tree Model	(59.52% - 64.39%)
Random Forest Model	(63.10% - 64.65%)

V. Discussion

We confirm that the goals of the research are achieved. All three main questions are answered and using Decision Tree and Random Forest models, we have built models capable of correctly classifying abalone into four classes using other features.

Decision tree models are nonparametric meaning the features used are not pre-set in advance and the model would

use a greedy method to find the best ones given the outcomes of Gini impurity [8]. Usage of Gini was preferred given better execution speed. This would give the decision tree the susceptibility to overfitting the training data [5], [14]. As such, I used both pruning and max depth regularization to ensure the model generalization error is kept under check.

From the above intervals, we suggest using the random forest as the model generates more reliable test accuracy scores with a lower standard deviation that boosts certainty. We note that random forests are more likely to perform better in reducing generalization error than a single decision tree by reducing the variance at the marginal costs of higher bias [10]. However, if ease of explanation is of crucial importance, I suggest using the decision tree model [11], [13] since the sacrifice in accuracy by about 2% justifies the benefits of drawing a graphical tree.

In the case of random forest models, we have higher certainty in results as the standard deviation is only about a quarter of the decision tree model. At the same time, decision tree model accuracy scores from test and train are close with 63.85% and 61.95% for train and test average accuracies respectively. This is not the case with random forest models when train and test accuracy scores have over 10% discrepancy, suggesting the random forest model could be slightly overfitting.

It is worth mentioning that the strategy I used in finding the best model in each case was sequentially tuning hyperparameters. In other words, in the case of random forest, I first did a grid search from 1 to 1001 on several bagged trees and as I compared the test performance, I selected the best one and then shifted the focus to finding the max depth by manually testing few scenarios and last, with CCP Alpha, I did two grid searches. The other strategy instead could be doing all three simultaneously using nested for loops in Python (e.g., 3 in this case). Surely this grid search would be more comprehensive, resource-demanding yet more reliable and thorough.

Furthermore, related to limitations, it should be noted that we could do a multi-dimensional grid search instead of a sequential one and find the best combination for the tree numbers, pruning alpha and max depth concerning the test accuracy simultaneously. This would surely provide us with a better set of hyperparameters given a more thorough search. The main reason I did not do this search using nested for loops in Python was primarily due to the computational power and time.

Finally, applying principal component analysis to first reduce dimensions may greatly help with reducing the number of features in the decision tree model. The random forest model automatically performs feature selection, yet the decision tree model is more likely to struggle here and reducing 8 features to only a few principal components could help with improving model runtime speed.

VI. CONCLUSION

In conclusion, I built and tested a few models using decision trees and random forests to classify abalone into four age categories using 8 features ("Sex" was transformed into 3 columns using one-hot encoding). Model accuracies suggest one could expect 60-65% accuracy when applying them to new abalone data.

Furthermore, related to future research, researchers may want to apply neural networks or boosting e.g., XGBoost using related software packages such as Keras, Sci-kit, and xgboost to develop respective models for the classification task. Boosting is one of the main ideas in learning [11], [15] and has a great deal to contribute to classification tasks such as this. At the same time, it should be noted that as the case with the Decision Tree model vs Random Forest one, if explaining how the model has come to generate the respective result is of importance, then applying neural networks or boosting the ensemble could also remain a challenge.

Last, multi-attribute hyperparameter search could improve models' performance and should access to the hardware is within reach, they may want to consider tuning 3 hyperparameters concerning each other simultaneously.

VII. REFERENCES

[1] Joshua Starmer, "Jupyter Notebook: Classification Trees in Python from Start to Finish," StatQuest!!! https://statquest.org/product/jupyter-notebook-classification-trees-in-python-from-start-to-finish (accessed Jun. 14, 2022)

[2] J. R. Quinlan, "Decision trees and decision-making," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 2, pp. 339-346, March-April 1990, DOI: 10.1109/21.52545.

[3] R. Polikar, "Ensemble Learning," Ensemble Machine Learning, pp. 1–34, 2012, DOI: 10.1007/978-1-4419-9326-7_1.

[4] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001, DOI: 10.1023/a:1010933404324.

[5] Aurélien Géron, Hands-on machine learning with Scikit-Learn and TensorFlow concepts, tools, and techniques to build intelligent systems. O'Reilly Media, Inc., 2019

[6] P. Cerda and G. Varoquaux, "Encoding High-Cardinality String Categorical Variables," in IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 3, pp. 1164-1176, 1 March 2022, DOI: 10.1109/TKDE.2020.2992529.

[7] G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning. New York, NY: Springer New York, 2013. DOI: 10.1007/978-1-4614-7138-7.

[8] Rohitash Chandra, "ZZSC5836 (H3 2022) Ed — Digital Learning Platform, Data Mining, and Machine Learning" edstem.org. https://edstem.org/au/courses/8454/lessons/20313/slides/144660 (accessed Jun. 14, 2022)

[9] A. Olteanu, "Tutorial: Learning Curves for Machine Learning in Python," Dataquest, Jan. 03, 2018. https://www.dataquest.io/blog/learning-curves-machine-learning (accessed Jun. 18, 2022).

[10] Rohitash Chandra, "ZZSC5836 (H3 2022) Ed — Digital Learning Platform, Data Mining, and Machine Learning" edstem.org. https://edstem.org/au/courses/8454/lessons/21236/slides/150764 (accessed Jun. 17, 2022).

[11] T. Hastie, R. Tibshirani, and J. Friedman, The elements of statistical learning, second edition: data mining, inference, and prediction. New York: Springer, 2009.

[12] L. Breiman, "Bagging predictors," Machine Learning, vol. 24, no. 2, pp. 123–140, Aug. 1996, DOI: 10.1007/bf00058655.

[13] Oleg Okun, Giorgio Valentini, and Matteo Re, Ensembles in machine learning applications / 3rd Workshop on Supervised and Unsupervised Ensemble Methods and their Applications, Barcelona, Spain, 2010. Berlin: Springer-Verlag, 2011.

[14] M. Kuhn, K. Johnson, and Springer Science+Business Media Applied predictive modelling. New York: Springer, 2016.

[15] J. Brownlee, "Essence of Bootstrap Aggregation Ensembles," Machine Learning Mastery, May 16, 2021. https://machinelearningmastery.com/essence-of-bootstrap-aggregation-ensembles/