

Predicting Tree Growth Models Using Machine Learning

[Chow Jie Seth](#)








Table of Contents

1. Introduction.....	2
2. Proposed Methodology.....	3
3. Implementation.....	4
3.1. MTG to CSV File Conversion.....	4
3.2. Extracting RGR Values from XML files.....	4
3.3. Data Preprocessing.....	5
3.4. Classification Training.....	6
3.5. Classification Prediction.....	8
3.6. Regression Training.....	9
3.7. Regression Prediction.....	10
4. Conclusion.....	11
5. Future Work.....	13
References.....	14

1. Introduction

Our research aims to predict tree growth in Singapore's urban environment [1] using just a single snapshot of remote sensing data. The development of this predictive model allows us to efficiently plan tree maintenance without the need for physical periodic inspections. As a result, this approach could significantly reduce costs, manpower, and resources required for tree maintenance, making a valuable contribution to sustainable urban development and greener cities. Ultimately, our work has the potential to promote a more eco-friendly and efficient approach to tree management in Singapore and other regions.

The MTG files used in this study are node-based datasets converted from point cloud data, which contain essential information like the tree's radius. By leveraging machine learning (ML) techniques, we aim to predict the growth model (Figure 1) of a given tree based on its MTG file data and subsequently determine its growth model parameters. In this paper, we will delve into the application of machine learning for identifying the tree growth models and its corresponding parameters. This research represents a significant step forward in urban forestry practices, where technology meets sustainability for the betterment of our cities and the environment.

Name	Form	Model dM/dt	Biomass M time basis	AGR dM/dt time basis	RGR (dM/dt)/ M time basis	RGR (dM/dt)/ M mass basis
Linear		r	$M_0 + rt$	r	$\frac{r}{M_0 + rt}$	$\frac{r}{M}$
Exponential		rM	$M_0 e^{rt}$	$rM_0 e^{rt}$	r	r
Power law		rM^β	$(M_0^{1-\beta} + rt(1-\beta))^{1/(1-\beta)}$	$r(M_0^{1-\beta} + rt(1-\beta))^{\beta/(1-\beta)}$	$r(M_0^{1-\beta} + rt(1-\beta))^{-1}$	$rM^{\beta-1}$
Monomolecular		$r(K - M)$	$K - e^{-rt}(K - M_0)$	$re^{-rt}(K - M_0)$	$\frac{r(K - M_0)}{M_0 + K(e^{rt} - 1)}$	$\frac{r(K - M)}{M}$
Three-parameter logistic		$rM \left(1 - \frac{M}{K}\right)$	$\frac{M_0 K}{M_0 + (K - M_0)e^{-rt}}$	$\frac{rM_0 K e^{-rt}(K - M_0)}{(M_0 + e^{-rt}(K - M_0))^2}$	$\frac{re^{-rt}(K - M_0)}{M_0 + e^{-rt}(K - M_0)}$	$r \left(1 - \frac{M}{K}\right)$
Four-parameter logistic*		$r(M - L) \left(\frac{K - M}{K - L}\right)$	$L + \frac{M_0(K - L)}{M_0 + P e^{-rt}}$	$\frac{rM_0(K - L)P e^{-rt}}{(M_0 + P e^{-rt})^2}$	$\frac{rM_0(K - L)P e^{-rt}}{(M_0 + P e^{-rt})(L P e^{-rt} + M_0 K)}$	$\frac{r(M - L)}{M} \left(\frac{K - M}{K - L}\right)$
Gompertz		$rM \left(\ln \frac{K}{M}\right)$	$K \left(\frac{M_0}{K}\right)^{e^{-rt}}$	$rK e^{-rt} \left(\frac{M_0}{K}\right)^{e^{-rt}} \ln \frac{K}{M_0}$	$re^{-rt} \ln \frac{K}{M_0}$	$r \ln \frac{K}{M}$

* $P = K - L - M_0$.

Figure 1. Known Growth Models of Trees [2]

2. Proposed Methodology

In order to utilise ML to predict the tree growth model (Figure 1) of the trees in the dataset, we propose a methodological approach. Before inputting the datasets into the ML models, we must first perform some data engineering on the raw information of the MTG and XML files of the tree datasets.

Firstly, we convert the MTG files into CSV format for more convenient data handling in Python. Additionally, from the XML files, we extract the RGR parameter information for these trees.

Next, after cleaning and preprocessing the datasets, we can proceed to train and test classification ML models using datasets of trees with known growth models. Using the most accurate model, we will predict the growth model label of the unseen datasets containing trees with unknown growth models.

Similarly, for predicting the corresponding parameters of the tree growth models, we will train and test regression ML models on trees with known parameter values. Subsequently, we will predict the parameter values of the unseen dataset using the model with the best accuracy.

For the regression task, it is essential to use the dataset of trees with known parameters that share the same growth model label as the unseen dataset to achieve accurate predictions.

Addressing the following **assumptions** in the proposed methodology:

1. The assumption that the diameter growth rate of the tree follows the growth model of the tree. The relationship between the diameter growth rate of the tree and its other characteristics, such as parent and child node diameters, remains consistent with a particular growth model.
2. The assumption that the growth model of the tree remains constant throughout the project duration, with periodic updates to accommodate any changes as the tree grows.
3. The assumption that external stimuli, such as sunlight and rain, will not significantly affect the growth of the tree.

3. Implementation

3.1. MTG to CSV File Conversion

The MTG files contain additional information, such as classes and descriptions, that are not required for our purposes. Therefore, when converting from MTG to CSV files, we can simply omit this unnecessary information. We will start from the line in the MTG files with the nodes and split it into its elements, which consist of "Entity-Code," "XX," "YY," and "ZZ" values (Figure 2).

By appending each of these lines into a list, we can later convert each line to a row in a CSV file for more straightforward data handling. This process allows us to retain only the relevant information needed for our machine learning analysis, ensuring a more efficient and effective approach to predicting tree growth models and parameters.

ENTITY-CODE	YY	XX	radius	ZZ
/I	0	0	0	0
^<I	-0.00162018527778	-0.0105702168392	0.305817236212	0.0882196743379
^<I	-0.00323230866551	-0.0210878372703	0.306095178711	0.176445782397
^<I	-0.0048364098906	-0.0315531204782	0.30637444117	0.264678260155

Figure 2. Snippet of the Node Data

3.2. Extracting RGR Values from XML files

To ensure the correctness of the RGR values for each tree, we perform a filtering process on both the XML and MTG files. This filtering ensures that each XML and MTG file label matches correctly.

Subsequently, from the XML files, we locate the <diameter_growth_rate type="RGR"> tag and extract the string of numbers that follows it (Figure 3).

```
<diameter_growth_rate type="RGR">0.433049760467</diameter_growth_rate>
```

Figure 3. Snippet of the RGR tag in a XML file

In the extracted string of numbers from the XML files, each digit represents a value on a scale of 0 to 9. This scale is used to encode the percentage of a range of known minimum and maximum parameter values that the tree can have, based on its specific tree growth model.

For instance, if a digit in the string has a value of 0, it corresponds to the minimum value of the parameter range, while a digit with a value of 9 corresponds to the maximum value of the parameter range.

Additionally, the position of each digit in the string corresponds to a specific parameter for the tree growth model. Thus, the first digit represents parameter 'r', the second digit parameter 'beta', and so on (Table 1.).

Digit Position	Corresponding Parameter
1st	r
2nd	beta
3rd	K
4th	L

Table 1. Digit Position and Corresponding Parameters

3.3. Data Preprocessing

Before preprocessing the data, we create a function that maps the trees in the dataset to their known growth models for training the classification models. Trees with unknown growth models are labelled as 'unknown'.

Next, we calculate the diameter of each node using the extracted data from the MTG files. Additionally, we employ a stack data structure to track the parent nodes of each node and save this information in a dataframe with columns 'node_ID' and 'parent_ID'. Here, the parent node of

a given node is defined as the node immediately connected to it from the bottom of the tree. By leveraging these IDs, we can map each node to its corresponding parent node's diameter.

Then, we flatten each tree, organising each node's diameter along with its corresponding parent node's diameter in a 2D array for each tree. These arrays are combined into a dataframe, with each row representing a tree (Figure 4). Finally, we concatenate the data frames containing tree node diameters with their respective growth models and parameter values.

1	node_1_diameter	node_1_parent_diameter	node_2_diameter	node_2_parent_diameter	node_3_diameter	node_3_parent_diameter
2	0	0	0.611634472	0	0.612190357	0.611634472
3	0	0	0.552065312	0	0.55245894	0.552065312
4	0	0	0.632161251	0	0.632739041	0.632161251
5	0	0	0.618080996	0	0.618509179	0.618080996
6	0	0	0.553474256	0	0.554121575	0.553474256
7	0	0	0.64023338	0	0.640742896	0.64023338
8	0	0	0.699562658	0	0.70005025	0.699562658
9	0	0	0.685561701	0	0.685909289	0.685561701
10	0	0	0.530340987	0	0.530742061	0.530340987

Figure 4. Screenshot of the First 3 Child-Parent Node Diameters of the First 10 Trees

3.4. Classification Training

Classification Task - Predicting Growth Model Type of Trees:

The classification task in the provided code aims to predict the growth model type of trees. The input variables for this task are the child and parent node diameters, while the target variable is the growth model type of each tree. The code utilises multiple classifiers from Python's Scikit-learn (Sklearn) library to perform the prediction. The classifiers used in the code are as follows: Decision Trees, Random Forests, Naive Bayes, and K-Nearest Neighbours.

Data Splitting:

To evaluate the performance of the classifiers and avoid overfitting, the code utilises Sklearn's `train_test_split()` function to split the preprocessed dataset into training and testing samples. The dataset is divided into training samples, which consist of 80% of the data, and testing samples, which consist of the remaining 20% of the data. The samples with 'unknown' labels will be used as the unseen dataset for prediction. Stratified sampling is employed during the splitting process, ensuring that the proportion of different growth model types remains consistent in both the training and testing datasets (Figure 5).

```
Number of classes in dataset:
growth_model_type
linear          565
exponential     202
monomolecular   101
power_law       100
unknown         86
Name: count, dtype: int64
```

```
Number of classes in training dataset:
growth_model_type
linear          372
exponential     161
monomolecular    81
power_law        80
Name: count, dtype: int64
```

```
Number of classes in testing dataset:
growth_model_type
linear          93
exponential     41
monomolecular   20
power_law       20
Name: count, dtype: int64
```

Figure 5. Screenshot of the Composition of the Number of Samples in the Dataset and the Training and Testing Dataset

Training and Evaluation:

The code then proceeds to train each classifier on the training dataset. Each classifier learns from the relationship between the input features (child and parent node diameters) and the corresponding growth model types. Once the training is complete, the classifiers are used to predict the growth model types for the unseen data in the testing dataset.

Model Evaluation:

After making predictions on the testing dataset, the code calculates the accuracy and generates a classification report for each classifier (Figure 6). The classification report provides additional evaluation metrics such as precision, recall, and F1-score for each growth model type, offering a comprehensive view of the models' performance.

Decision Tree

Accuracy: 0.867816091954023				
Classification Report:				
	precision	recall	f1-score	support
exponential	0.82	0.80	0.81	41
linear	0.89	0.92	0.91	93
monomolecular	0.95	0.90	0.92	20
power_law	0.78	0.70	0.74	20
accuracy			0.87	174
macro avg	0.86	0.83	0.84	174
weighted avg	0.87	0.87	0.87	174

Accuracy: 0.5402298850574713				
Classification Report:				
	precision	recall	f1-score	support
exponential	0.00	0.00	0.00	41
linear	0.54	1.00	0.70	93
monomolecular	0.00	0.00	0.00	20
power_law	1.00	0.05	0.10	20
accuracy			0.54	174
macro avg	0.38	0.26	0.20	174
weighted avg	0.40	0.54	0.38	174

Random Forest

Accuracy: 0.9022988505747126				
Classification Report:				
	precision	recall	f1-score	support
exponential	0.91	0.78	0.84	41
linear	0.87	1.00	0.93	93
monomolecular	1.00	0.95	0.97	20
power_law	1.00	0.65	0.79	20
accuracy			0.90	174
macro avg	0.95	0.85	0.88	174
weighted avg	0.91	0.90	0.90	174

Accuracy: 0.8563218390804598				
Classification Report:				
	precision	recall	f1-score	support
exponential	0.82	0.78	0.80	41
linear	0.83	0.98	0.90	93
monomolecular	1.00	1.00	1.00	20
power_law	1.00	0.30	0.46	20
accuracy			0.86	174
macro avg	0.91	0.76	0.79	174
weighted avg	0.87	0.86	0.84	174

Naive Bayes

K-Nearest Neighbours

Figure 6. Screenshot of the Accuracy and Classification Report of the Respective ML Models

Choosing the Best Model:

The code determines the best-performing classifier based on its accuracy score. The classifier with the highest accuracy is selected as the best model for predicting the growth model types of trees. As we can observe, the Random Forest model achieves the best accuracy, and it is chosen as the best-performing classifier.

3.5. Classification Prediction

The selected Random Forest classifier is then used to predict the growth model types of unseen tree data, consisting of 86 tree samples (Figure 5). The prediction results for the unseen dataset are presented in Figure 7. It is observed that all the tree samples in the unseen dataset are predicted to belong to the "linear" growth model type.


```
[ 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear' 'linear'
  'linear' 'linear' 'linear' 'linear' 'linear' 'linear' ]
```

Figure 7. Results from Classification Prediction

3.6. Regression Training

Regression Task - Predicting Growth Model Parameters of Trees:

The regression task in the provided code aims to predict the parameters of trees based on their child and parent node diameters. The target variable for this task is the parameter 'r' for each tree. Since the predicted growth model is linear, only one parameter needs to be predicted.

However, for trees with other growth models, the code may require multiple runs or a loop to predict additional parameters such as 'beta' and 'K'. Also, the code employs two regression algorithms from Python's Scikit-learn (Sklearn) library for the parameter prediction task:

Decision Trees and Random Forests.

Dataset Splitting:

To train the regression models, the code removes the nonlinear trees from the dataset. As shown in Figure 5, the dataset consists of 372 linear tree samples used for training and 93 samples for testing.

Model Evaluation:

To evaluate the accuracy of the regression models, the code calculates the percentage difference between the predicted 'r' value and the actual 'r' value in the testing dataset. If the percentage difference is within 10%, the predicted 'r' value is considered accurate. The final accuracy of the model is then calculated as the number of accurate predictions divided by the total number of tested samples (Figure 8).

Decision Tree

Accuracy: 35.483870967741936

Random Forest

Accuracy: 9.67741935483871

Figure 8. Screenshot of the Accuracies of the Respective ML Models

3.7. Regression Prediction

As we can observe from the accuracy results in Figure 8, the decision tree regressor is much more accurate than the random forest regressor. Thus, we selected the decision tree regressor for predicting the unseen dataset of 86 samples. The prediction results are presented in Figure 9.

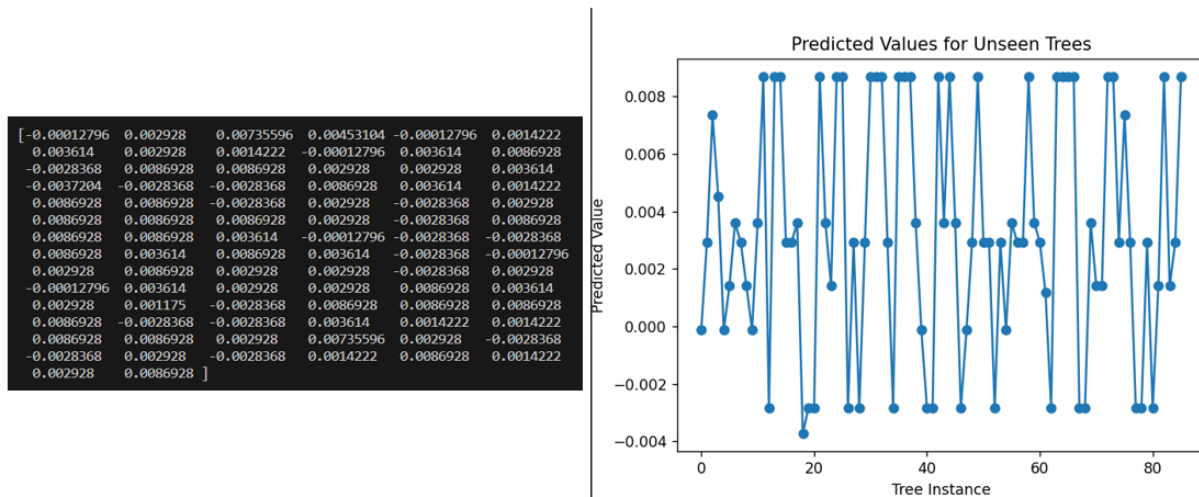


Figure 9. Results from Regression Prediction

4. Conclusion

The classification models, which focused on predicting the growth model types of trees based on child and parent node diameters, demonstrated promising accuracy levels (Figure 10). On the other hand, the regression models, which aimed to predict specific parameters such as 'r',

showed relatively lower accuracy (Figure 11). This discrepancy in performance suggests that the chosen regression approach might not be well-suited for the given dataset and task.

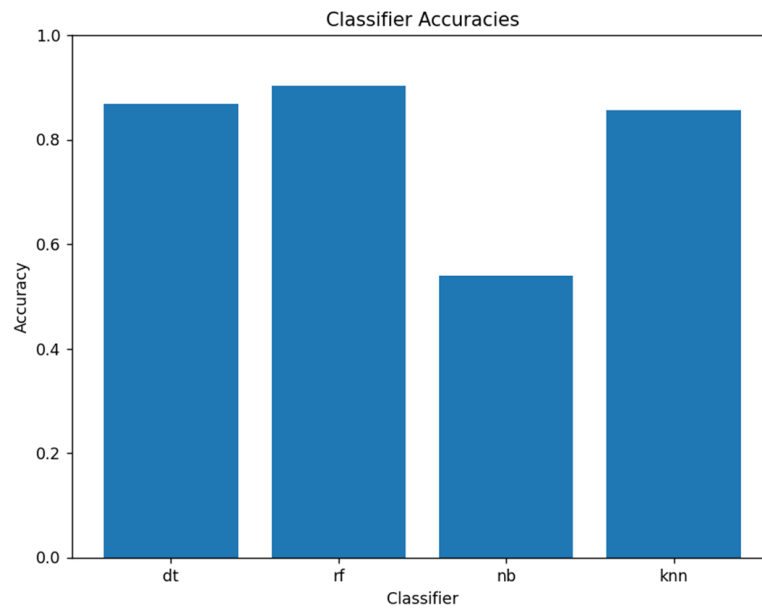


Figure 10. Accuracies of the Classifiers in a Bar Graph Plot

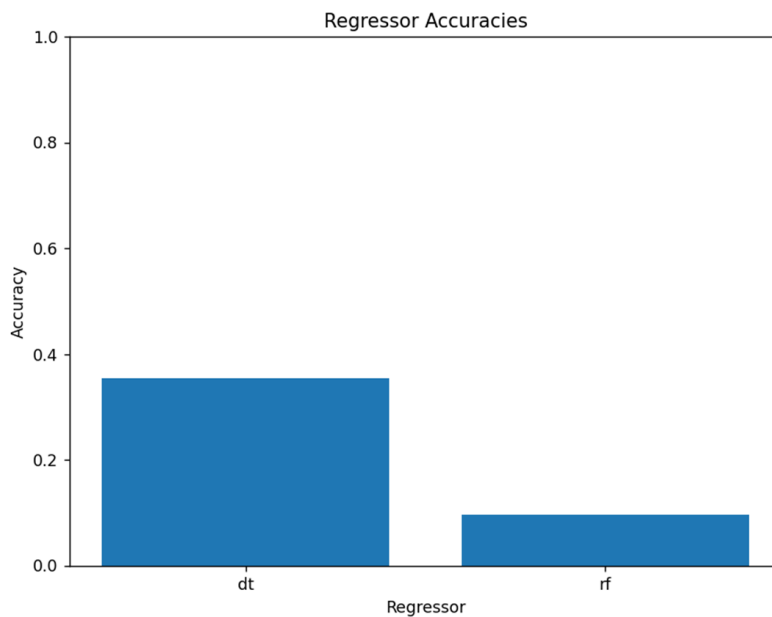


Figure 11. Accuracies of the Regressors in a Bar Graph Plot

We will now discuss the strengths and weaknesses of the project.

Strengths:

1. The project explores various classification models such as Decision Trees, Random Forests, Naive Bayes, and K-Nearest Neighbours. This allows for a comprehensive evaluation of different algorithms and helps identify the most suitable classifier for the task.
2. The use of stratified sampling during the train-test split ensures that the distribution of classes in both the training and testing datasets remains representative of the original dataset. This approach prevents bias and improves the reliability of model evaluation.
3. The project introduces a custom accuracy metric for the regression task, considering the percentage difference between predicted and actual 'r' values. This tailored metric provides a more meaningful evaluation of the regression models' performance, especially when the task involves predicting continuous values.

Weaknesses:

1. Currently, the regression task utilises only child and parent node diameters as input features. Incorporating additional relevant features related to tree characteristics, such as species, external environmental factors, and diameter ratios, could potentially enhance the prediction accuracy and provide a more comprehensive understanding of the growth patterns.
2. The project does not implement hyperparameter tuning for the regression models. Conducting hyperparameter tuning for the regression models could lead to improved accuracy and generalisation on unseen data.
3. The accuracy of the predictions might be affected by any underlying bias present in the training dataset. If the training dataset is imbalanced or contains biased samples, it could lead to biased predictions and reduced generalisation to unseen data.
4. The project currently employs a train-test split for model evaluation. It would be beneficial to explore other validation strategies, such as cross-validation, to assess

model performance more robustly. Cross-validation can provide a better estimate of the model's generalisation capabilities and reduce the risk of overfitting.

5.Future Work

In order to enhance the predictive capabilities of the model and address the limitations observed in the current implementation, several future works and improvements can be considered. First, it is worth exploring the integration of deep learning algorithms, such as Convolutional Neural Networks (CNNs), to effectively capture intricate patterns and relationships in tree structures. Additionally, expanding feature engineering to include more relevant attributes related to tree growth, such as environmental factors and soil conditions, can provide the model with more comprehensive information for accurate predictions. Addressing class imbalance in the training dataset is essential, as it can improve the model's ability to handle different growth model types effectively. By addressing these aspects, the model can be further refined to achieve more advanced and accurate predictions, benefiting various fields, including forestry management, environmental research, and urban planning.

Acknowledgements. I would like to express my heartfelt gratitude to my mentor, Dr. Like Gobeawan, from the Institute of High Performance Computing (IHPC), A*STAR, for her guidance throughout this internship. I am also grateful to IHPC for providing me with the opportunity to work on this project and gain valuable learning experience.

References

1. Alonso. (2021, January 13). Success story: the transformation of Singapore into a sustainable garden city. <https://tomorrow.city/a/singapore-transformation-garden-city>
2. Timothy Paine, R. Marthews, R. Vogt, Purves, Rees, Hector, & A. Turnbull. (2011, September 29). Methods in Ecology and Evolution. <https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.2041-210X.2011.00155.x>