IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 6, June 2024

DOI: 10.17148/IARJSET.2024.11624

AID362 Bioassay Classification and Regression (Neuronal Network and Extra Tree) with Machine Learning

Sowmya T[1], Sandeep S[2], Chirag Suresh[3], Rachana N[4], Vasantha Poojary[5]

Assistant Professor, Computer Science and Engineering, BMS College of Engineering, Bengaluru, India[1]

Assistant Professor, Electronics and Communication Engineering, BMS College of Engineering, Bengaluru, India[2]

Student, Computer Science and Engineering, BMS College of Engineering, Bengaluru, India[3]

Student, Computer Science and Engineering, BMS College of Engineering, Bengaluru, India[4]

Student, Computer Science and Engineering, BMS College of Engineering, Bengaluru, India[5]

Abstract: In order to anticipate and classify bioassay data labeled AID362, this study investigates the use of sophisticated machine learning techniques, including neural networks and extra trees. The usefulness and suitability of these algorithms for bioassay categorization and regression are examined by means of meticulous testing and analysis. The UCI repository dataset is prepared by applying several data preprocessing techniques such as cleaning, normalization, and feature scaling. Overfitting is reduced and dimensionality is made easier with the help of Extra Trees feature extraction. The effectiveness of the trained models in managing high-dimensional, nonlinear bioassay data is demonstrated by a variety of performance indicators. The findings add significant knowledge to the field by demonstrating the effectiveness of neural networks in identifying complex patterns and the advantages of extra trees in managing complicated datasets.

Keywords: Bioassay, AID362, Classification, Regression.

I. INTRODUCTION

Pharmaceutical and toxicological research relies heavily on the evaluation of chemical substances' biological activity and toxicity. The AID362 dataset is a useful resource that includes detailed data from several bioassays carried out on chemical substances. Molecular descriptors, activity classifications, and toxicity levels are only a few of the many aspects that these assays cover.

In toxicological and pharmaceutical research, bioassays are essential for assessing the toxicity and biological activity of chemical substances. Bioassay data analysis has typically involved manual labor and a lengthy processing period. The field has experienced a revolution with the integration of machine learning techniques, which provide automated solutions that improve accuracy and speed up analysis. The main focus of this work is the AID362 dataset, a repository with a wealth of data on different types of bioassays carried out on chemical substances. Our goal is to efficiently handle bioassay classification and regression tasks by utilizing the capabilities of machine learning methods, particularly Neuronal Networks and Extra Trees.

Neuronal Networks: Inspired by the neural networks seen in the human brain, these systems are able to recognize complex patterns and correlations in data. Bioassay analysis, which involves intricate relationships between chemical characteristics and biological responses, is a suitable application for them due to their versatility and ability to do nonlinear mapping.Extra Trees are an ensemble learning technology that work in tandem with neural networks to enhance predictive performance. They build several decision trees and integrate their outputs. Strong solutions for bioassay analysis are provided by Extra Trees, who are especially skilled at managing high-dimensional data and reducing noise.Our goals in doing this research project are to apply the Neuronal Network and Extra Tree models, assess their effectiveness with the AID362 dataset, and determine the advantages and disadvantages of each strategy through a comparison analysis. In order to improve our knowledge of chemical compound behavior and toxicity and ultimately aid in the creation of safer and more potent medications, this work aims to shed light on the use of machine learning in bioassay analysis.

II. LITERATURE SURVEY

A summary of bioassay analysis is as follows: bioassay analysis measures the impact of substances on living things or biological systems in order to determine their biological activity. This can involve evaluating toxicity levels, figuring out how potent medications are, and finding possible treatment agents.

Bioassay Classification and Regression Using Machine Learning : The application of machine learning techniques in bioassay analysis has grown in recent years because of their capacity to process enormous datasets and discern intricate patterns. In order to forecast a compound's biological activity and toxicity based on its chemical features, classification and regression methods are frequently used.

Neuronal Networks: With their interconnected nodes arranged in layers, neural networks are modeled after the organization of the human brain. Bioinformatics and drug development are only two of the many fields that use them because of their ability to learn intricate correlations between input and output variables.

Extra Trees: Extra Trees is an ensemble learning technique that builds several decision trees and aggregates the predictions to increase precision. It works especially well with high-dimensional data, and noisy or incomplete datasets can be handled with ease.

A review of related research on machine learning algorithms for bioassay classification and regression is presented. The literature includes studies on deep learning-based structure-activity relationship modeling for chemical toxicity classification and the use of decision trees, random forest, and deep neural networks for predicting acute oral toxicity.

III. IMPLEMENTATION

Data Collection and Preprocessing of the dataset

- Description of AID362 Dataset: The AID362 dataset contains information on various bioassays conducted on chemical compounds. It includes features such as molecular descriptors, activity classes, and toxicity levels.
- Data Cleaning: The dataset underwent cleaning procedures to remove missing values, outliers, and redundant features. This ensures the quality and integrity of the data used for model training.
- Feature Selection: Feature selection techniques were employed to identify the most relevant features for bioassay analysis. This helps improve model performance and reduce computational complexity.
- Data Normalization: Normalization techniques were applied to scale the input features to a standard range. This ensures that all features contribute equally to the model training process and prevents biases towards certain variables.
- Data Acquisition : The data for the project is acquired from the UCI repository, as described in the literature
- Data Preprocessing : The acquired data is preprocessed to make it ready for further processing, including data cleaning, normalization, and feature scaling.
- Feature Extraction : Relevant features are extracted to reduce the data set's feature space and prevent model overfitting. The Extra Tree ensemble feature selection technique is used to retrieve the optimal feature set.
- Model Training : The Neuronal Network and Extra Tree models are trained using the preprocessed data and the extracted features.
- Model Evaluation : The performance of the models is evaluated using metrics such as accuracy, F1-score, precision, recall, and computation time.

The architecture of the neural network model was developed utilizing input, hidden, and output layers, which are three different layers of neurons. A range of optimization strategies and activation functions were investigated in an effort to enhance model performance.
Extra Tree technique: An ensemble of decision trees was constructed using the Extra Trees technique. For optimal results, parameters like the maximum depth and tree count were optimized.
Training and Evaluating the Models: Using metrics like accuracy, precision, recall, and F1-score, the preprocessed dataset was used to train the Neuronal Network and Extra Tree models. To guarantee robustness and generalizability, cross-validation techniques were utilized.

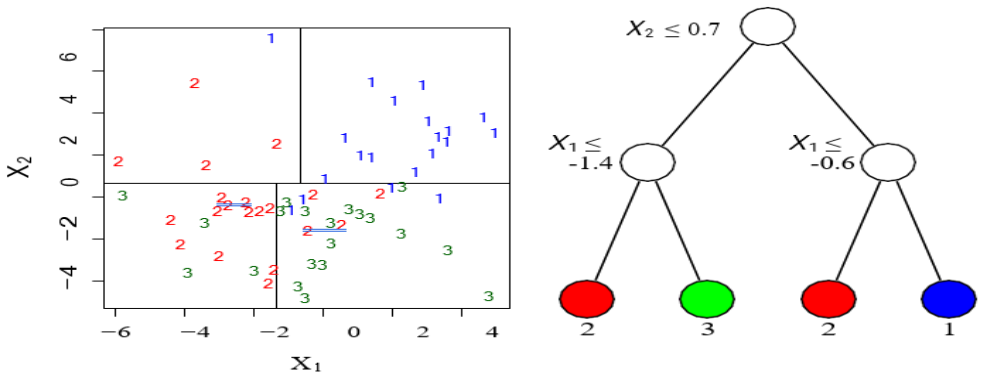


Figure 1: Partitions (left) and decision tree structure (right) for a CART classification model with three classes labeled 1, 2, 3. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. The predicted class is given beneath each leaf node.

A training sample of n observations on a class variable Y that takes values 1, 2,..., k, and p predictor variables, X1,..., Xp, is available to us in a classification issue. Finding a model to forecast Y values from fresh X values is our aim. Theoretically, the answer is as easy as splitting the X divide the space into k distinct sets, A1, A2,..., Ak, so that, for j = 1, 2,..., k, the expected value of Y is j if X belongs to Aj. Two traditional methods for handling ordered values in the X variables are nearest-neighbor classification [13] and linear discriminant analysis [17]. These methods yield sets *Aj* with piecewise linear and nonlinear, respectively, boundaries that are not easy to interpret if *p* is large.

Classification tree approaches split the data set recursively, one X variable at a time, producing rectangle sets Aj. This facilitates interpretation of the sets. As an illustration, consider the scenario with three classes and two X variables shown in Figure 1. The relevant decision tree structure is displayed in the right panel, while the data points and divisions are plotted in the left panel. The tree structure has the advantage of being applicable to an unlimited number of variables, while the plot on its left can only be used to a maximum of two.
The first published classification tree algorithm is THAID [16, 35]. Employing a mea- sure of node impurity based on the distribution of the observed Y values in the node, THAID splits a node by exhaustively searching over all X and S for the split X S that minimizes the total impurity of its two child nodes. If X takes ordered values, the set S is an interval of the form (, c]. Otherwise, S is a subset of the values taken by X. The process is applied recursively on the data in each child node. Splitting stops if the relative decrease in impurity is below a pre-specified threshold. Algorithm 1 gives the pseudocode for the basic steps.

```
Algorithm 1 Pseudocode for tree construction by exhaustive search
1. Start at the root node
2. For each X, find the set S that minimizes the sum of the node impurities in the two child nodes and choose the split X∗ S∗that gives the minimum over all X and S.
3. If a stopping criterion is reached, exit. Otherwise, apply step 2 to each
child node in turn.
```

C4.5 [38] and CART [5] are two later classification tree algorithms that follow this approach. C4.5 uses entropy for its impurity function while CART uses a generaliza- tion of the binomial variance called the Gini index. Unlike THAID, however, they first grow an overly large tree and then prune it to a smaller size to minimize an estimate of the misclassification error. CART employs ten-fold (default) cross-validation while C4.5 uses a heuristic formula to estimate error rates. CART is implemented in the R system [39] as RPART [43], which we use in the examples below.Despite its simplicity and elegance, the exhaustive search approach has an undesirable property. Note that an ordered variable with m distinct values has (m 1) splits of the form X c, and an unordered variable with m distinct unordered values has (2^{m-1} 1) splits of the form X S. Therefore if everything else is equal, variables that have more distinct values have a greater chance to be selected. This selection bias affects the integrity of inferences drawn from the tree structure.Building on an idea that originated in the FACT [34] algorithm, CRUISE [21, 22], GUIDE [31] and QUEST [33] use a two-step approach based on significance tests to split each node. First, each X is tested for association with Y and the most significant variable is selected. Then an exhaustive search is performed for the set S. Because every X has the same chance to be selected if each is independent of Y , this approach is effectively free of selection bias. Besides, much computation is

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁒ Peer-reviewed & Refereed journal ⁒ Vol. 11, Issue 6, June 2024

DOI: 10.17148/IARJSET.2024.11624

saved as the search for S is carried out only on the selected X variable. GUIDE and CRUISE use chi-squared tests, and QUEST uses chi-squared tests for unordered and analysis of variance tests for ordered variables. CTree [18], another unbiased method, uses permutation tests. Pseudocode for the GUIDE algorithm is given in Algorithm 2. The CRUISE, GUIDE and QUEST trees are pruned the same way as CART.

Algorithm 2 Pseudocode for GUIDE classification tree construction

```
1. Start at the root node.
2. For each ordered variable X, convert it to an unordered variable X′ by grouping its values in the node into a small number of intervals. If X is unordered, set X′ = X.
3. Perform a chi-squared test of independence of each X′ variable versus Y on the data in the node and compute its significance probability.
4. Choose the variable X∗ associated with the X′ that has the smallest significance probability
5. Find the split set X∗S∗that minimizes the sum of Gini indexes and use it to split the node into two child nodes.
6. If a stopping criterion is reached, exit. Otherwise, apply steps 2–5 to each child node.
7. Prune the tree with the CART method.
```

CHAID [20] employs yet another strategy. If X is an ordered variable, its data values in the node are split into ten intervals and one child node is assigned to each interval. If X is unordered, one child node is assigned to each value of X. Then CHAID uses significance tests and Bonferroni corrections to try to iteratively merge pairs of child nodes. This approach has two consequences. First, some nodes may be split into more than two child nodes. Second, owing to the sequential nature of the tests and the inexactness of the corrections, the method is biased toward selecting variables with few distinct values.CART, CRUISE and QUEST can allow splits on linear combinations of all the ordered variables while GUIDE can split on combinations of two variables at a time. If there are missing values, CART and CRUISE use alternate splits on other variables when needed, C4.5 sends each observation with a missing value in a split through every branch using a probability weighting scheme, QUEST imputes the missing values lo- cally, and GUIDE treats missing values as belonging to a separate category. All except C4.5 accept user-specified misclassification costs and all except C4.5 and CHAID ac- cept user-specified class prior probabilities. By default, all algorithms fit a constant model to each node, predicting Y to be the class with the smallest misclassification cost. CRUISE can optionally fit bivariate linear discriminant models and GUIDE can fit bivariate kernel density and nearest neighbor models in the nodes. GUIDE also can produce ensemble models using bagging [3] and random forest [4] techniques. Table 1 summarizes the features of the algorithms.To see how the algorithms perform in a real application, we apply them to a data set on new cars for the 1993 model year [26]. There are 93 cars and 25 variables. We let the Y variable be the type of drive train, which takes three values (rear, front, or four-wheel drive). The X variables are listed in Table 2. Three are unordered (manuf, type, and airbag, taking 31, 6, and 3 values, respectively), two binary-valued (manual and domestic), and the rest ordered. The class frequencies are rather unequal: 16 (17.2%) are rear, 67 (72.0%) are front, and 10 (10.8%) are four-wheel drive vehicles. To avoid randomness due to ten-fold cross-validation, we use leave-one-out (i.e., n- fold) crossvalidation to prune the CRUISE, GUIDE, QUEST, and RPART trees in this article.Figure 2 shows the results if the 31-valued variable manuf is excluded. The CHAID tree is not shown because it has no splits. The wide variety of variables selected in the splits is due partly to differences between the algorithms and partly to the absence of a dominant X variable. Variable passngr is chosen by three algorithms (C4.5, GUIDE QUEST); enginsz, fuel, length, and minprice by two; and hp, hwympg,

Table 1 Comparison of classification tree methods. A check mark indicates pres- ence of a feature. The codes are: b = missing value branch, c = constant model, d= discriminant model, i = missing value imputation, k = kernel density model, l = linear splits, m = missing value category, n = nearest neighbor model, u = univariate splits, s = surrogate splits, w = probability weights.

DOI: 10.17148/IARJSET.2024.11624

w = probability weights.

Feature	C4.5	CART	CHAID	CRUISE	GUIDE	QUEST
Unbiased splits				√	√	√
Split type	*u*	*u,l*	*u*	*u,l*	*u,l*	*u,l*
Branches/split	≥2	2	≥2	≥2	2	2
Interaction tests					√	

√

Pruning √ √ √ √ √ User-specified costs √ √ √ √ √ User-specified priors
√ √ √ √

Variable ranking		√			√	
Node models	*c*	*c*	*c*	*c,d*	*c,k,n*	*c*
Bagging & ensembles					√	
Missing values	*w*	*s*	*b*	*i,s*	*m*	*i*

luggage, maxprice, rev, type, and width by one each. Variables airbag, citympg, cylin, midprice, and rpm are not selected by any.When GUIDE does not find a suitable variable to split a node, it looks for a linear split on a pair of variables. One such split, on enginsz and rseat, occurs at the node marked with an asterisk (*) in the GUIDE tree. Restricting the linear split to two variables allows the data and the split to be displayed in a plot, as shown in Figure 3. Clearly, no single split on either variable alone can do as well in separating the two classes there.

Figure 4 shows the C4.5, CRUISE and GUIDE trees when variable manuf is included. Now CHAID, QUEST and RPART give no splits. Comparing them with their coun- terparts in Figure 2, we see that the C4.5 tree is unchanged, the CRUISE tree has an additional split (on manuf), and the GUIDE tree is much shorter. This behavior is not uncommon when there are many variables with little or no predictive power: their introduction can substantially reduce the size of a tree structure and its prediction ac- curacy; see, e.g., [15] for more empirical evidence.Table 3 reports the computational times used to fit the tree models on a computer with a 2.66Ghz Intel Core 2 Quad Extreme processor. The fastest algorithm is C4.5, which takes milliseconds. If manuf is excluded, the next fastest is RPART, at a tenth of a second. But if manuf is included, RPART takes more than three hours—a 105-fold increase. This is a practical problem with the CART algorithm: because manuf takes 31 values, the algorithm must search through 230 1 (more than 1 billion) splits at the root node alone. (If Y takes only two values, a computational shortcut [5, p. 101] reduces the number of searches to just 30 splits.) C4.5 is not similarly affected because it does not search for binary splits on unordered X variables. Instead, C4.5 splits the node into one branch for each X value and then merges some branches after the tree.

Table 2 Predictor variables for the car data

Variable	Description	Variable	Description
manuf	Manufacturer (31 values)	rev	Engine revolutions per mile
type	Type (small, sporty, compact,	manual	Manual transmission
	midsize, large, van)		available (yes, no)
minprice	Minimum price (in $1,000)	fuel	Fuel tank capacity (gallons)
midprice	Midrange price (in $1,000)	passngr	Passenger capacity (persons)
maxprice	Maximum price (in $1,000)	length	Length (inches)
citympg	City miles per gallon	whlbase	Wheelbase (inches)
hwympg	Highway miles per gallon	width	Width (inches)
airbag	Air Bags standard (0, 1, 2)	uturn	U-turn space (feet)
cylin	Number of cylinders	rseat	Rear seat room (inches)
enginzs	Engine size (liters)	luggage	Luggage capacity (cu. ft.)
hp	Maximum horsepower	weight	Weight (pounds)
rpm	Revolutions per minute	domestic	Domestic (U.S./non U.S.
	at maximum horsepower		manufacturer)

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁔ Peer-reviewed & Refereed journal ⁔ Vol. 11, Issue 6, June 2024

DOI: 10.17148/IARJSET.2024.11624

Table 3 Tree construction times on a 2.66Ghz Intel Core 2 Quad Ex- treme processor for the car data.

C4.5	CRUIS E	GUID E	QUES T	RPA RT
Without 0.004s	manuf3.57s	2.49s	2.26s	0.09s
With manuf	0.003s4.00s	1.86s	2.54s	3h 2m

CRUISE, GUIDE and QUEST also are unaffected because they search ex- haustively for splits on unordered variables only if the number of values is small. If the latter is large, these algorithms employ a technique in [34] that uses linear discriminant analysis on dummy variables to find the splits.

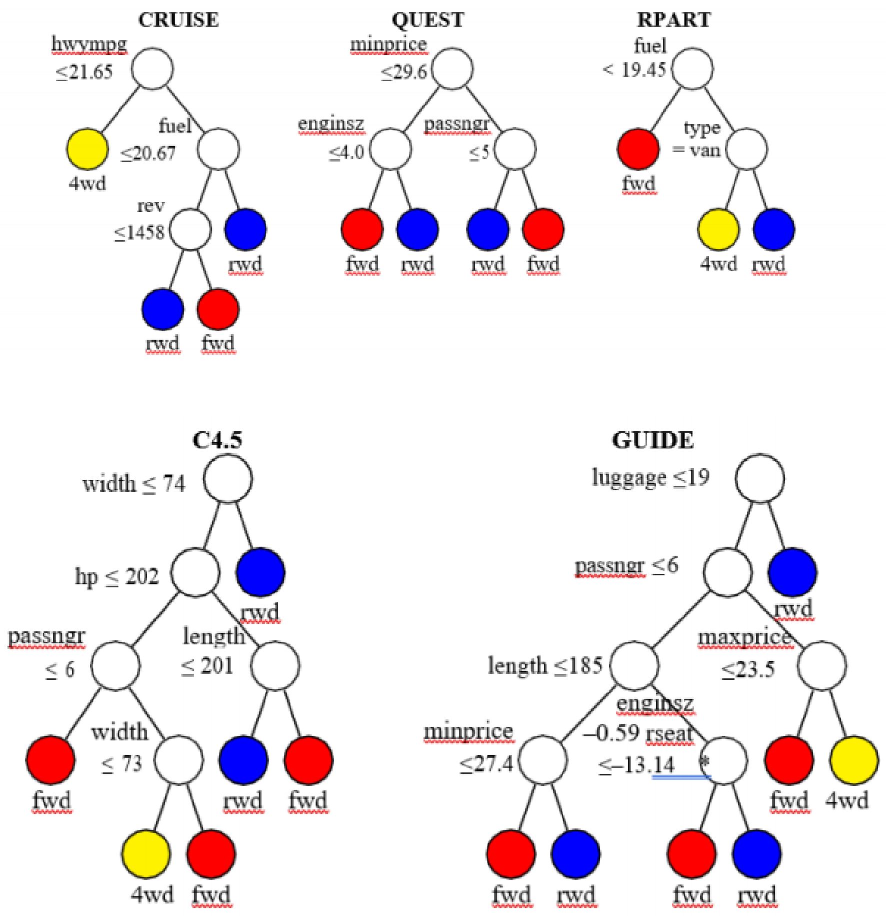


Figure 2: CRUISE, QUEST, RPART, C4.5 and GUIDE trees for car data without manuf. The CHAID tree is trivial with no splits. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. The predicted class is given beneath each leaf node .

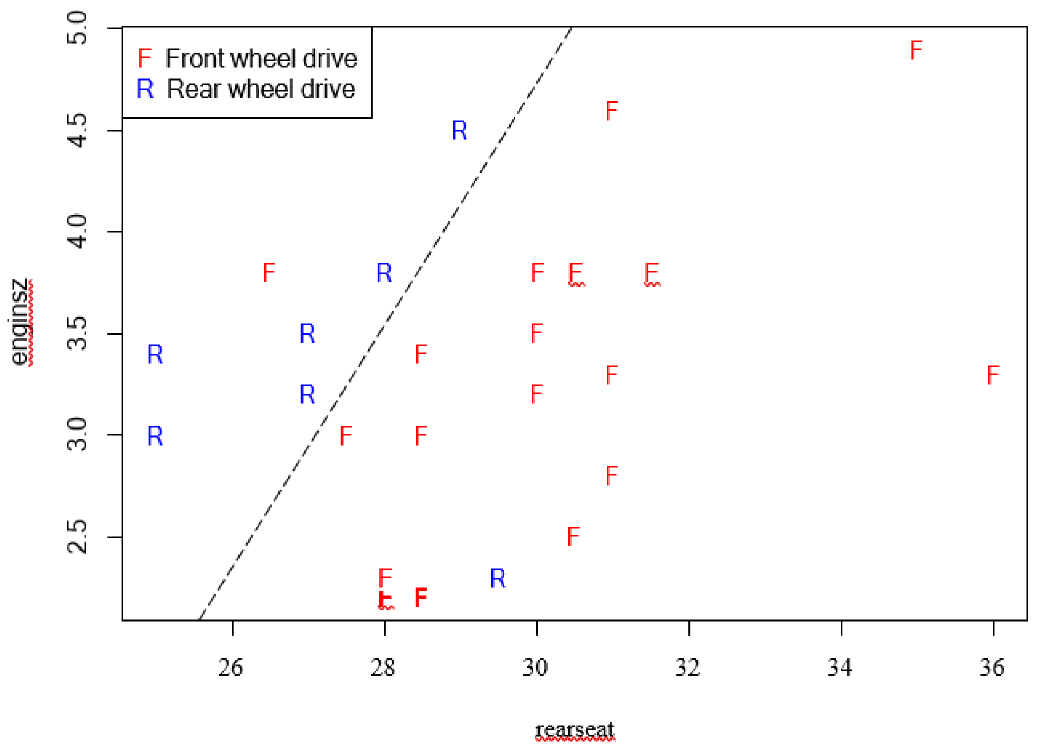


Figure 3: Data and split at the node marked with an asterisk (*) in the GUIDE tree in Figure 2.

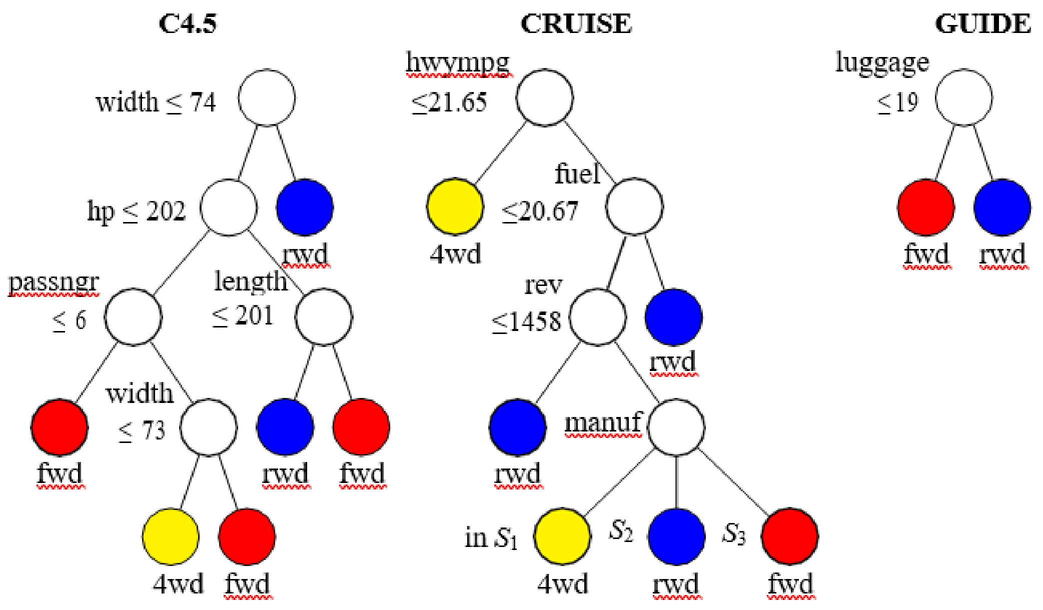
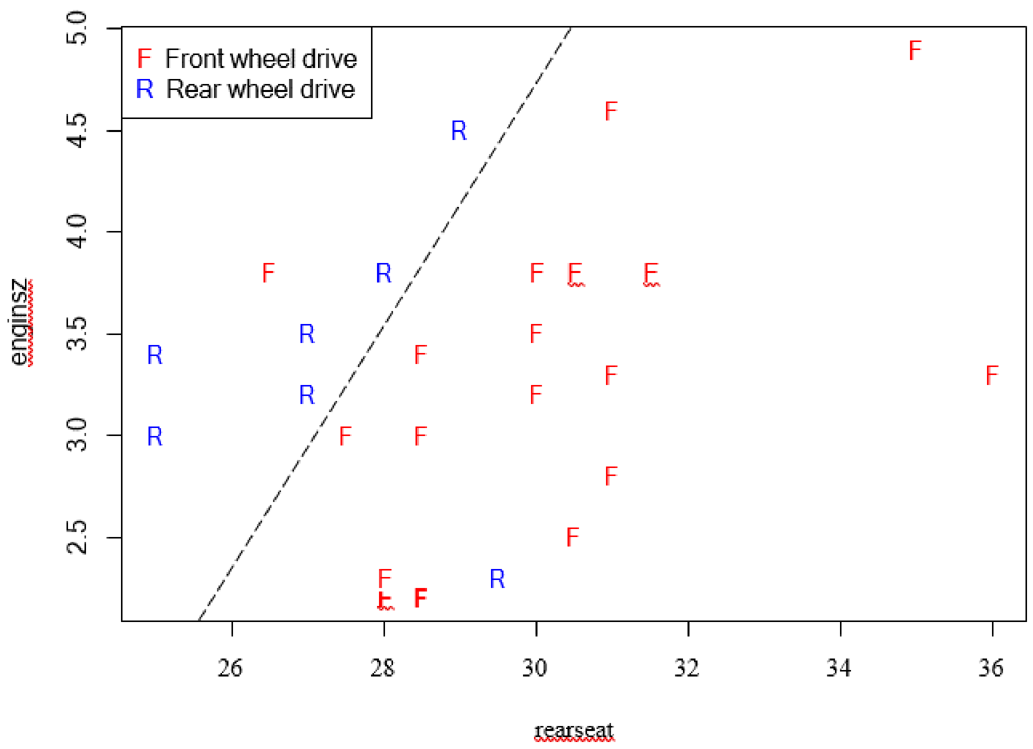


Figure 4: CRUISE, C4.5 and GUIDE trees for car data with manuf included. The CHAID, RPART and QUEST trees are trivial with no splits. Sets S1 and S2



Regression trees and classification trees are comparable, with the exception that the Y variable accepts ordered values. And each node is fitted with a regression model to get the expected values of Y. In the past, AID [36], which debuted a few years prior to THAID, was the first regression tree method. AID with the node prediction being the sample mean of Y, and CART regression tree methods according to Algorithm 1. The node impurity is the sum of squared deviations about the mean. Models with piecewise constants result from this. These models' prediction accuracy frequently falls short of more smoothly constructed models, despite their ease of interpretation.

It can be computationally demanding to apply this method to piecewise linear models, though, as each prospective split requires the fitting of two linear models, one for each child node. An variant of Quinlan's [37] regression tree approach, M5' [44], builds piecewise linear models using a more computationally effective method. It fits a linear regression model to the data in each leaf node after first building a piecewise constant tree. The resulting trees are typically larger than those from other piecewise linear tree approaches since the tree structure is the same as that of a piecewise constant model. GUIDE [27] addresses the regression problem by employing classification tree methods.

At each node, it fits a regression model to the data and computes the residuals. Then it defines a class variable Y ′ taking values 1 or 2, depending on whether the sign of the residual is positive or not. Finally it applies Algorithm 2 to the Y ′ variable to split the node into two. This approach has three advantages: (i) the splits are unbiased,
(ii) only one regression model is fitted at each node, and (iii) because it is based on residuals, the method is neither limited to piecewise constant models nor to the least squares criterion. Table 4 lists the main features of CART, GUIDE and M5’.

Table 4 Comparison of regression tree methods. A check mark indicates presence of a feature. The codes are: a = missing value category, c = constant model, g = global mean/mode imputation, l = linear splits, m = multiple linear model, p = polynomial model, r = stepwise linear model, s = surrogate splits, u = univariate splits, v = least squares, w= least median of squares, quantile, Poisson, and proportional hazards.

	Feature	CART	GUIDE	M5'
	Unbiased splits		√	
	Split type	*u,l*	*u*	*u*
	Branches/split	2	2	≥2
	Interaction tests	√	√	√
√	Pruning	√		
√	Variable importance ranking			
	Node models	*c*	*c,m,p,r*	*c,r*
	Missing value methods	*s*	*a*	*g*
	Loss criteria	*v*	*v,w*	*v*
	Bagging & ensembles		√	

To compare CART, GUIDE and M5' with ordinary least squares (OLS) linear regres- sion, we apply them to some data on smoking and pulmonary function in children [40]. The data, collected from 654 children aged 3–19, give the forced expiratory volume (FEV, in liters), gender (sex, M/F), smoking status (smoke, Y/N), age (years), and height (ht, inches) of each child. Using an OLS model for predicting FEV that in- cludes all the variables, Kahn [19] finds that smoke is the only one not statistically significant. He also finds a significant age-smoke interaction if ht is excluded, but not if ht and its square are both included. This problem with interpreting OLS models often occurs when collinearity is present (the correlation between age and height is 0.8). Figure 5 shows five regression tree models: (a) GUIDE piecewise constant (with sam- ple mean of Y as the predicted value in each node), (b) GUIDE best simple linear (with a linear regression model involving only one predictor in each node), (c) GUIDE best simple quadratic regression (with a quadratic model involving only one predictor in each node), (d) GUIDE stepwise linear (with a stepwise linear regression model in each node), and (e) M5' piecewise constant. The CART tree (from RPART) is a sub- tree of (a), with six leaf nodes marked by asterisks (*). In the piecewise polynomial models (b) and (c), the predictor variable is found independently in each node, and non-significant terms of the highest orders are dropped.

For example for model (b) in Figure 5, a constant is fitted in the node containing females taller than 66.2, because the linear term for ht is not significant at the 0.05 level. Similarly, two of the three leaf nodes in model (c) are fitted with first degree polynomials in ht because the quadratic terms are not significant. Since the nodes, and hence the domains of the polynomials, are defined by the splits in the tree, the estimated regression coefficients typically vary between nodes.Because the total model complexity is shared between the tree structure and the set of node models, the complexity of a tree structure often decreases as the complexity of the node models increases. Therefore the user can choose a model by trading off tree structure complexity against node model complexity. Piecewise constant models are mainly used for the insights their tree structures provide. But they tend to have low prediction accuracy, unless the data are are sufficiently informative and plentiful to yield a tree with many nodes. The trouble is that the larger the tree, the harder it is to derive insight from it. Trees (a) and (e) are quite large, but because they split almost exclusively on ht, we can infer from the predicted values in the leaf nodes that FEV increases monotonically with ht.The piecewise simple linear (b) and quadratic (c) models reduce tree complexity with- out much loss (if any) of interpretability. Instead of splitting the nodes, ht now serves exclusively as the predictor variable in each node. This suggests that ht has strong linear and possibly quadratic effects. On the other hand, the splits on age and sex point to interactions between them and ht. These interactions can be interpreted with the help of Figures 6 and 7, which plot the data values of FEV and ht and the fitted regression functions, with a different symbol and color for each node. In Figure 6, the slope of ht is zero for the group of females taller than 66.2 inches, but it is constant and non-zero across the other groups. This indicates a three-way interaction involving age, ht and sex. A similar conclusion can be drawn from Figure 7, where there are

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 6, June 2024
DOI: 10.17148/IARJSET.2024.11624

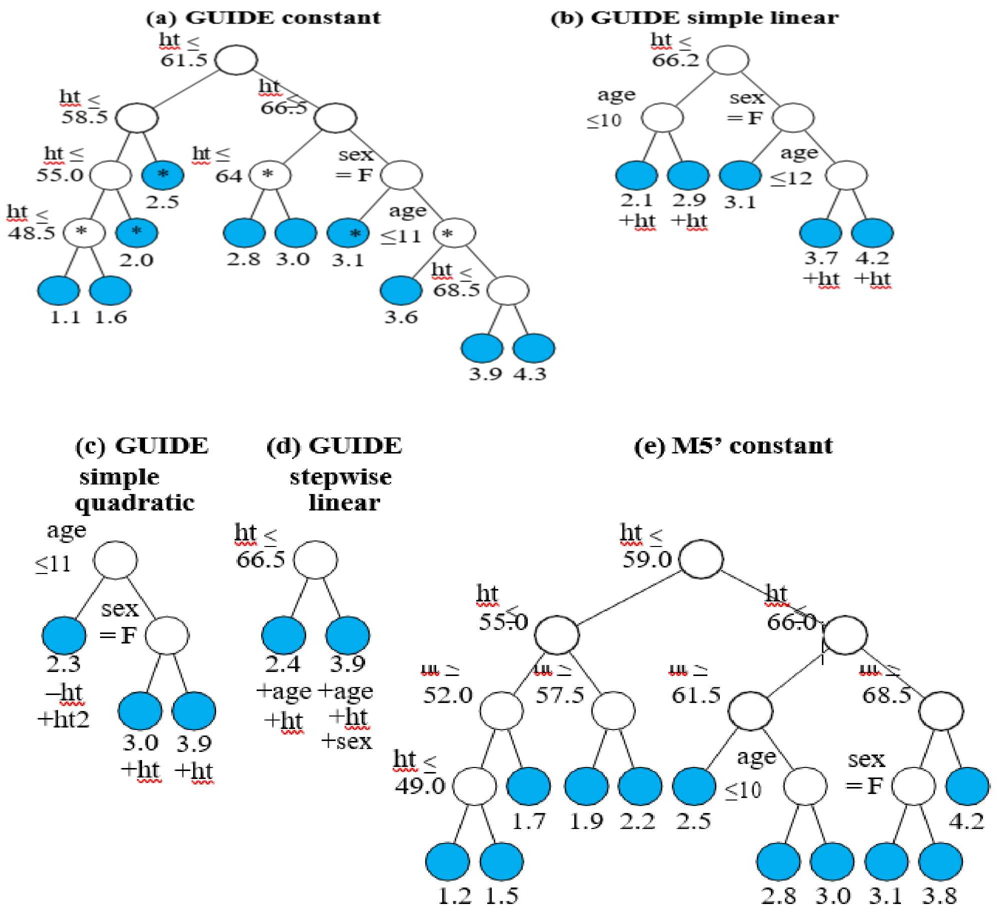


Figure 5: GUIDE piecewise constant, simple linear, simple quadratic, and stepwise linear, and M5' piecewise constant regression trees for predicting FEV. The RPART tree is a subtree of (a), with leaf nodes marked by asterisks (*). The mean FEV and linear predictors (with signs of the coefficients) are printed beneath each leaf node. Variable ht2 is the square of ht.

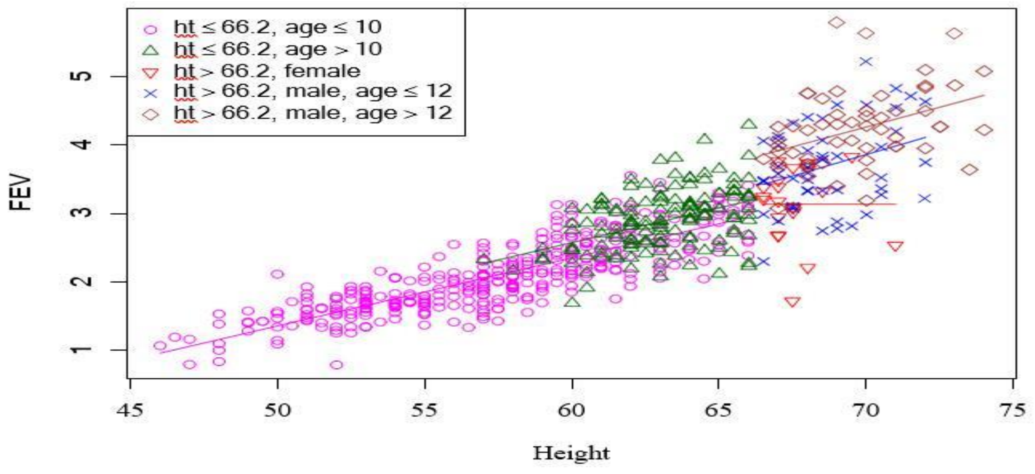


Figure 6: Data and fitted regression lines in the five leaf nodes of the GUIDE piecewise simple linear model in Figure 5(b).

DOI: 10.17148/IARJSET.2024.11624

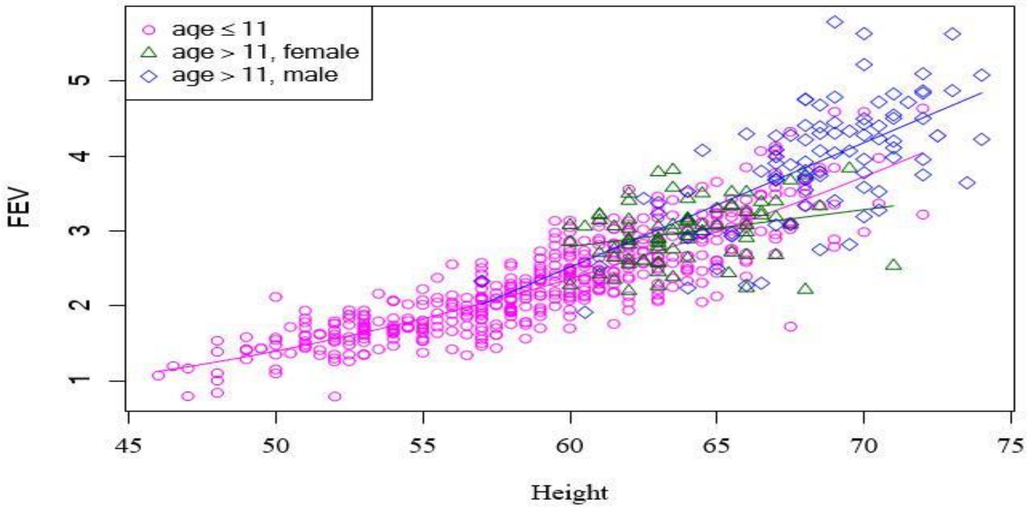


Figure 7: Data and fitted regression functions in the three leaf nodes of the GUIDE piecewise simple quadratic model in Figure 5(c)

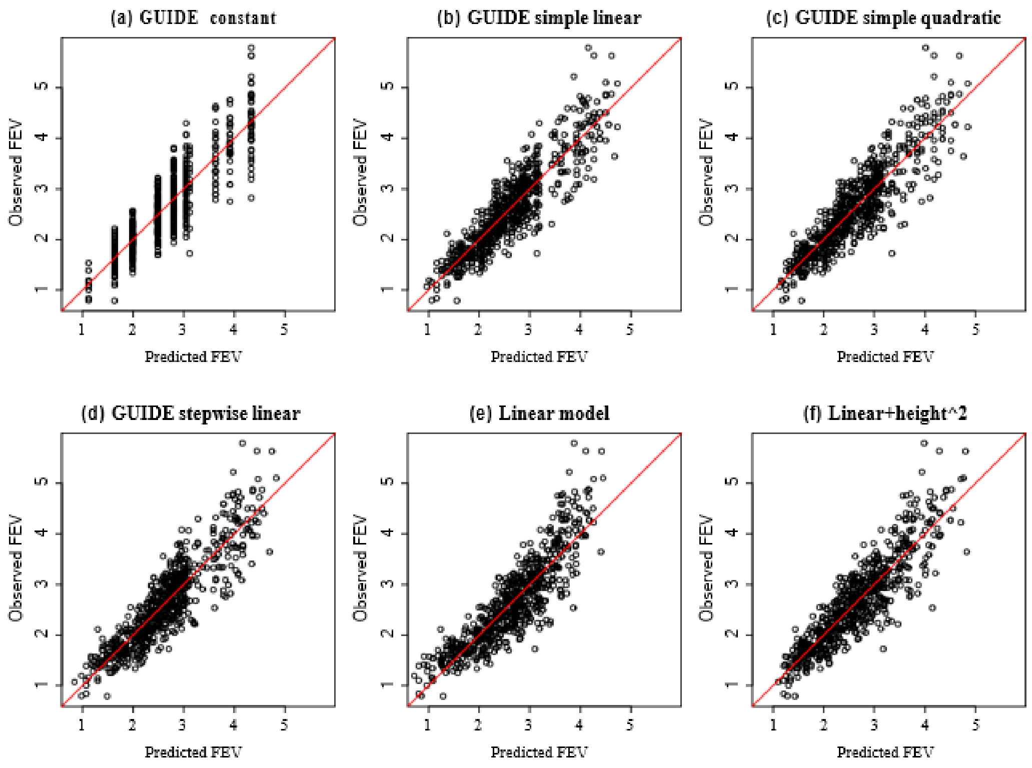


Figure 8: Observed versus predicted values for the tree models in Figure 5 and two ordinary least squares models.

only three groups, with the group of children aged 11 or below exhibiting a quadratic effect of ht on FEV. For children above age 11, the effect of ht is linear, but males have, on average, about a half liter more FEV than females. Thus the effect of sex seems to be due mainly to children older than 11 years. This conclusion is reinforced by the piecewise stepwise linear tree in Figure 5(d), where a stepwise linear model is fitted to each of the two leaf nodes. Age and ht are selected as linear predictors in both leaf nodes, but sex is selected only in the node corresponding to children taller than 66.5 inches, seventy percent of whom are above 11 years old. Figure 8 plots the observed versus predicted values of the four GUIDE models and two OLS models containing all the variables, without and with the square of height. The discreteness of the predicted values from the piecewise constant model is obvious, as is the curvature in plot (e). The piecewise simple quadratic model in plot (c) is strikingly similar to plot (f) where the OLS model includes ht-squared. This suggests that the two models have similar prediction accuracy. Model (c) has an advantage over model (f), however, because the former can be interpreted through its tree structure and the graph of its fitted function in Figure 7.

IV. RESULT

Performance Evaluation measures: Numerous measures, such as accuracy, precision, recall, and F1-score, were used to assess the performance of the Neuronal Network and Extra Tree models. The efficiency of each method in bioassay analysis was ascertained by comparing the results.

Analysis of the Neuronal Network and Extra Tree Models: Using a variety of bioassay classification and regression tasks, the models' performances were compared. Conclusions were made about the advantages and disadvantages of each algorithm for estimating biological activity and toxicity levels.

Interpretation of the outcomes: To find trends and patterns in the data, the experiment's outcomes were evaluated. The variables affecting the results of bioassays and the machine learning models' predictive power were shown.

V. CONCLUSION

In summary, this project has effectively demonstrated how Neuronal Networks and Extra Trees may be used for bioassay classification and regression problems. The algorithms have exhibited encouraging outcomes in precisely forecasting the biological activity and toxicity thresholds of chemical substances.

It is imperative to recognize specific constraints linked to these approaches. The performance and efficiency of both neural networks and extra trees can be affected by hyperparameter sensitivity and computational complexity. It is imperative to tackle these obstacles in order to improve the resilience and usefulness of these models.

Subsequent research projects may investigate different ensemble techniques and machine learning algorithms for bioassay analysis. Experimenting with several procedures can reveal which ones produce the most accurate and dependable predictions. The models' interpretability and reliability could be further improved by including domain knowledge and expert views into the modeling process, which would make them more appropriate for use in practical settings.

Through the resolution of these constraints and investigation of novel approaches for enhancement, scholars can propel the domain of bioassay analysis forward, ultimately playing a role in the creation of more efficacious and secure chemical compounds for diverse uses.

REFERENCES

[1]https://github.com/emirhanai/AID362-Bioassay-Classification-and-RegressionNeuronal-Network-and-Extra-Tree-with-Machine-Learnin/issues
[2]https://github.com/emirhanai/AID362-Bioassay-Classification-and-RegressionNeuronal-Network-and-Extra-Tree-with-Machine-Learnin
[3]https://www.frontiersin.org/journals/physiology/articles/10.3389/fphys.2019.01044/full
[4]https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9573053/5https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8885585/