Rule Fit Regressor

How it works:

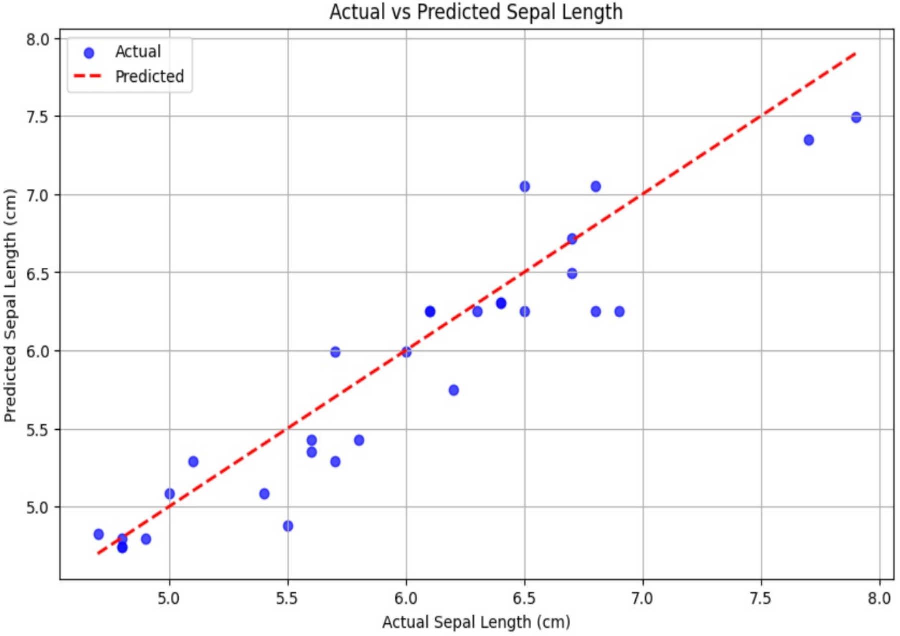
- Combines rules extracted from decision trees with linear terms for regression.
- Extracts interpretable decision rules from tree-based models.
- Uses these rules alongside linear terms to make predictions.
- Provides interpretability by allowing individual rules to be understood and analyzed.
- Effective in handling both categorical and continuous features.
- Suitable for tasks requiring a mix of linear and non-linear relationships.
- The RMSE for Rule Fit Regressor in this case is 0.338.

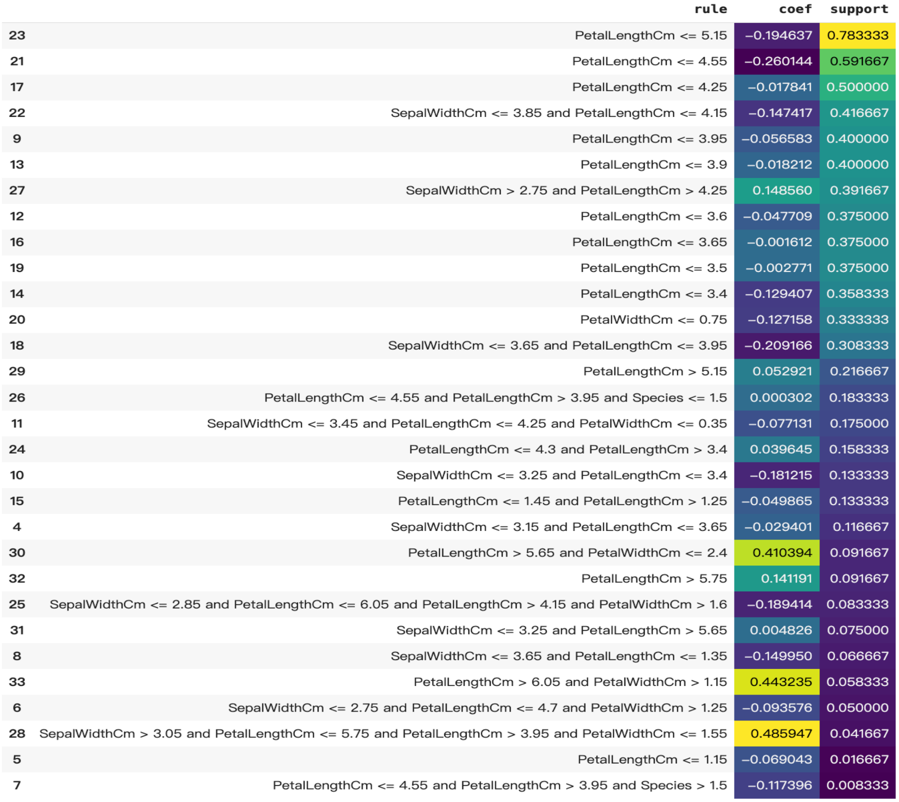
Pros:

- Adds feature interactions to linear models
- Rule fit works for both classification and regression
- Interpretability

Cons:

- Increasing number of features harms interpretability
- Overlapping rules cause difficulties in interpretation



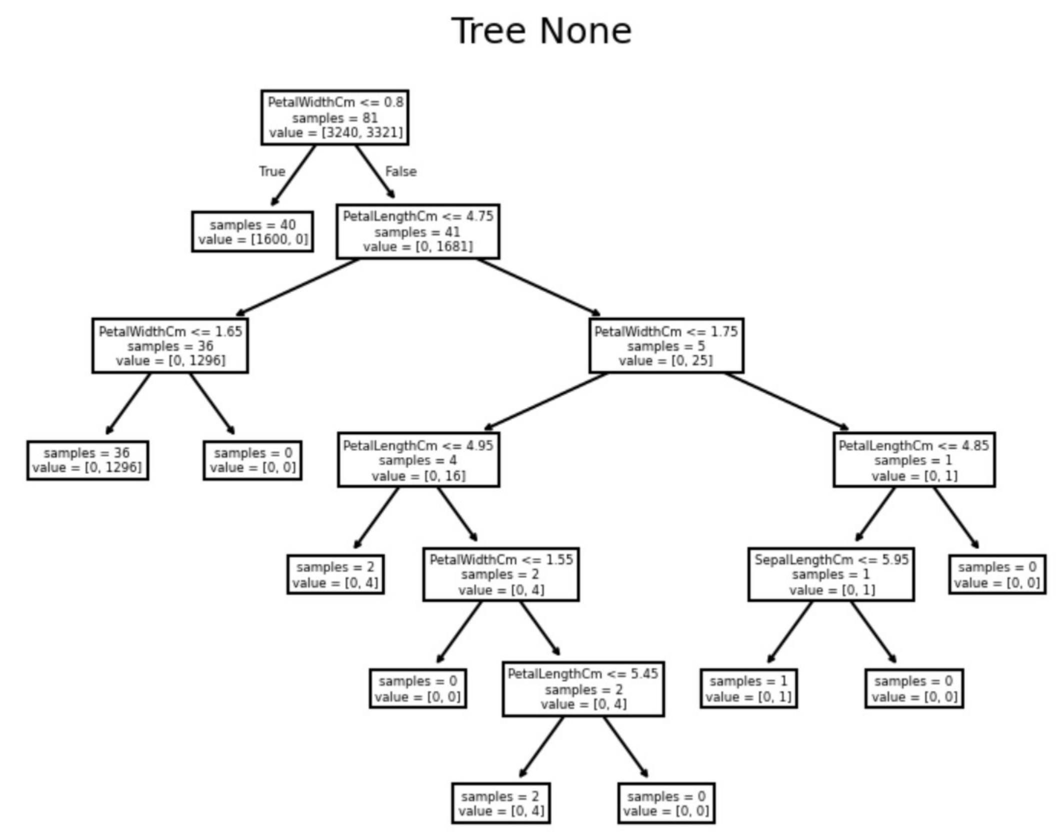


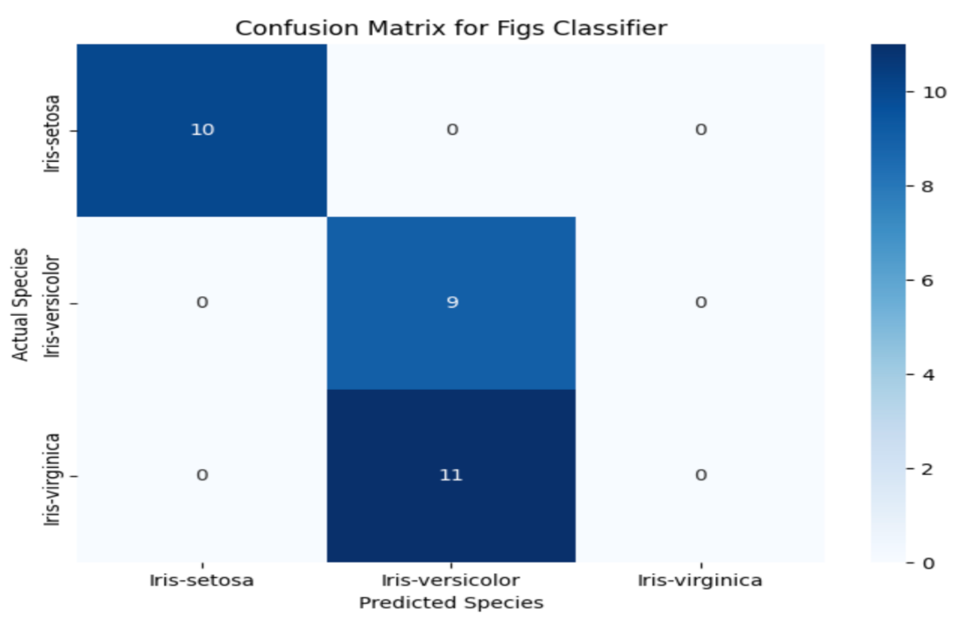
	rule	coef	support
23	PetalLengthCm <= 5.15	−0.194637	0.783333
21	PetalLengthCm <= 4.55	−0.260144	0.591667
17	PetalLengthCm <= 4.25	−0.017841	0.500000
22	SepalWidthCm <= 3.85 and PetalLengthCm <= 4.15	−0.147417	0.416667
9	PetalLengthCm <= 3.95	−0.056583	0.400000
13	PetalLengthCm <= 3.9	−0.018212	0.400000
27	SepalWidthCm > 2.75 and PetalLengthCm > 4.25	0.148560	0.391667
12	PetalLengthCm <= 3.6	−0.047709	0.375000
16	PetalLengthCm <= 3.65	−0.001612	0.375000
19	PetalLengthCm <= 3.5	−0.002771	0.375000
14	PetalLengthCm <= 3.4	−0.129407	0.358333
20	PetalWidthCm <= 0.75	−0.127158	0.333333
18	SepalWidthCm <= 3.65 and PetalLengthCm <= 3.95	−0.209166	0.308333
29	PetalLengthCm > 5.15	0.052921	0.216667
26	PetalLengthCm <= 4.55 and PetalLengthCm > 3.95 and Species <= 1.5	0.000302	0.183333
11	SepalWidthCm <= 3.45 and PetalLengthCm <= 4.25 and PetalWidthCm <= 0.35	−0.077131	0.175000
24	PetalLengthCm <= 4.3 and PetalLengthCm > 3.4	0.039645	0.158333
10	SepalWidthCm <= 3.25 and PetalLengthCm <= 3.4	−0.181215	0.133333
15	PetalLengthCm <= 1.45 and PetalLengthCm > 1.25	−0.049865	0.133333
4	SepalWidthCm <= 3.15 and PetalLengthCm <= 3.65	−0.029401	0.116667
30	PetalLengthCm > 5.65 and PetalWidthCm <= 2.4	0.410394	0.091667
32	PetalLengthCm > 5.75	0.141191	0.091667
25	SepalWidthCm <= 2.85 and PetalLengthCm <= 6.05 and PetalLengthCm > 4.15 and PetalWidthCm > 1.6	−0.189414	0.083333
31	SepalWidthCm <= 3.25 and PetalLengthCm > 5.65	0.004826	0.075000
8	SepalWidthCm <= 3.65 and PetalLengthCm <= 1.35	−0.149950	0.066667
33	PetalLengthCm > 6.05 and PetalWidthCm > 1.15	0.443235	0.058333
6	SepalWidthCm <= 2.75 and PetalLengthCm <= 4.7 and PetalWidthCm > 1.25	−0.093576	0.050000
28	SepalWidthCm > 3.05 and PetalLengthCm <= 5.75 and PetalLengthCm > 3.95 and PetalWidthCm <= 1.55	0.485947	0.041667
5	PetalLengthCm <= 1.15	−0.069043	0.016667
7	PetalLengthCm <= 4.55 and PetalLengthCm > 3.95 and Species > 1.5	−0.117396	0.008333

FIGS Classifier

How it works:

- Constructs decision rules through a greedy, interpretable process.
- Builds decision trees incrementally to minimize classification error.
- Each tree split maximizes the classification performance on a subset of data.
- Generates highly interpretable models with decision rules that can be easily understood.
- Effective in cases where interpretability is as important as accuracy.
- Produces compact models that can be efficiently evaluated.
- The accuracy for FIGS Classifier in this case is 0.633.



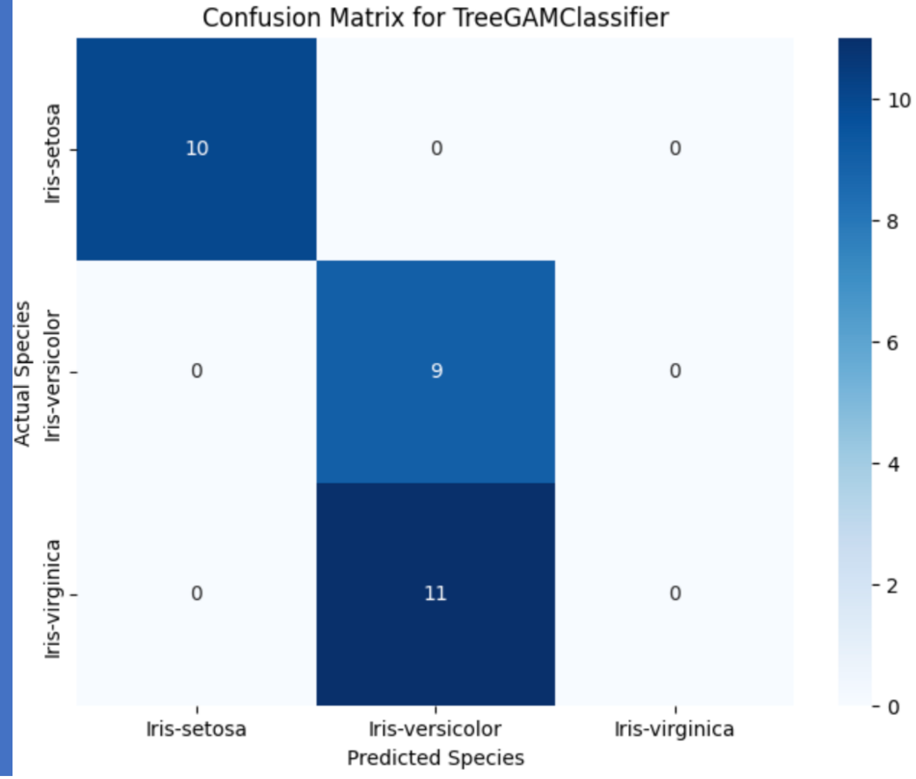


- **Root Node (PetalWidthCm <= 0.8)**: The first decision asks whether the petal width is less than or equal to 0.8. This splits the data into two branches:
  - **Left Branch (True)**: If PetalWidthCm <= 0.8, it leads to the left side, containing 40 samples from class 1.
  - **Right Branch (False)**: If PetalWidthCm > 0.8, it moves to the right, containing 41 samples that need further splitting.
- **Left Subtree**:
  - At the next split, PetalWidthCm <= 1.65, the samples are further split. If true, 36 samples are classified as class 1. If false, no samples remain in that branch.
- **Right Subtree**:
  - For samples where PetalWidthCm > 0.8, the next condition is whether PetalWidthCm <= 4.75. This branch is further split by various petal and sepal length/width conditions, classifying the remaining samples into either class 1 or class 2 based on those criteria.
- **Leaf Nodes**: At the bottom of the tree, each leaf node provides the final classification

Tree GAM Classifier

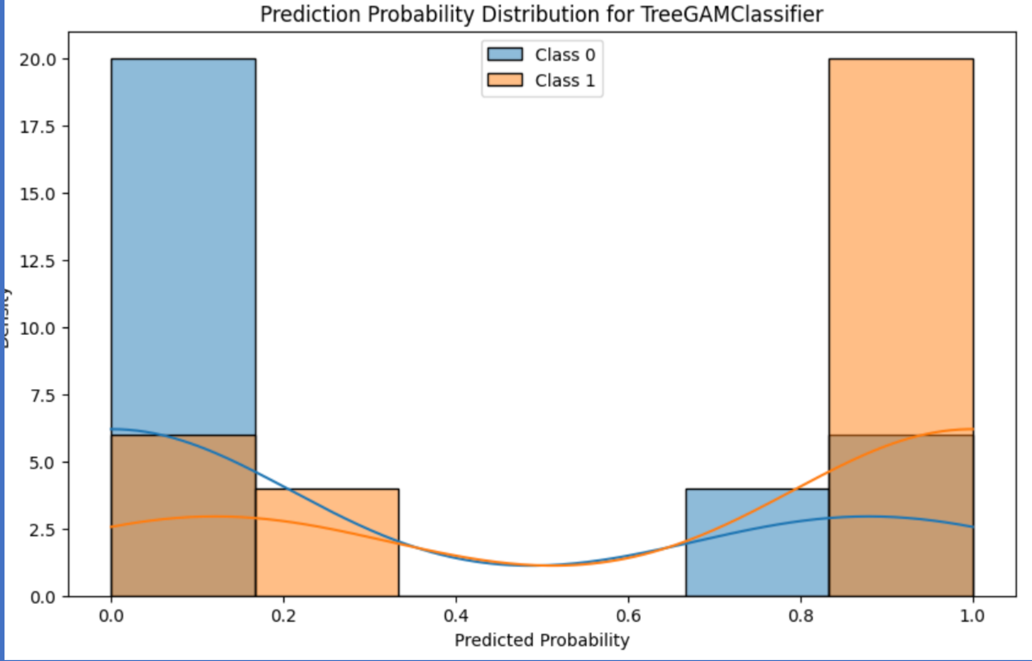
How it works:

- A tree-based Generalized Additive Model (GAM) for classification tasks.
- Combines tree-based models with additive components for flexibility and interpretability.
- Models the relationship between input variables and the target as a sum of non-linear functions.
- Captures non-linear relationships without sacrificing interpretability.
- Effective for datasets with complex, non-linear interactions.
- Suitable for tasks that require balancing accuracy and model interpretability.
- The accuracy for Tree GAM Classifier in this case is 0.633.



Prediction Probability Distribution for Tree GAM Classifier:

- Probability Distribution Peaks at 0 and 1:
  - For Class 0 (blue bars), we see that most of the predicted probabilities are concentrated at 0 and 1. This suggests that the model is confident in its predictions, assigning high or low probabilities (close to 0 or 1) rather than probabilities in the middle.
  - The same applies to Class 1 (orange bars), where most predictions are concentrated at probability values close to 1 (for Class 1) or close to 0 (for Class 0).
- Lower Density at Intermediate Values: There are very few instances where the model assigns probabilities between 0.4 and 0.6.



Comparison

Aspect	RuleFit Regressor	FIGS Classifier	TreeGAM Classifier
Type	Regression	Classification	Classification
Interpretability	High: Combines decision rules with linear terms	High: Builds interpretable decision trees through greedy process	High: Interpretable additive tree-based model
Core Algorithm	Rules extracted from decision trees and linear models	Greedy decision tree splitting	Generalized Additive Model (GAM) with tree-based components
Handles Non-linearity	Yes: Rules capture non-linear relationships	Yes: Trees naturally capture non-linear relationships	Yes: Additive components handle complex, non-linear patterns
Key Strengths	- Interpretable combination of rules and linear terms - Can handle both categorical and continuous features	- Produces compact, interpretable decision trees - Efficient and accurate	- Captures non-linear interactions - Balances flexibility and interpretability
Weaknesses	- Limited to regression tasks - Complexity increases with many rules	- Might not capture highly complex relationships without deep trees	- May require careful tuning for very high-dimensional data
Best Use Cases	- Regression tasks where interpretability is critical	- Classification tasks with a need for clear decision paths	- Classification with non-linear relationships between variables
Model Complexity	Moderate: Complexity grows with more decision rules	Low to Moderate: Generates compact decision trees	Moderate: Can handle non-linearities but remains interpretable