Baye-Tree: Integrating Naive Bayes into Decision Trees

Gus Simanson, Lalit Boyapati, and Anmol Karan
Thomas Jefferson High School for Science and Technology
Dr. Yilmaz
February 10, 2025

Abstract

Decision trees and Naive-Bayes are two of the most well-known and commonly used machine learning algorithms. However, each poses its own benefits and setbacks, which can be addressed to form a more robust and accurate hybrid machine learning model. Decision trees are strong in handling feature interactions and splitting data into low-entropy subsets. Naive Bayes, based on Bayes' theorem, is fast and efficient and relies on independence between features. While each has considerable advantages, they can be combined to be more suitable for high-dimensional datasets and for multi-class classification. Decision trees often suffer from overfitting, particularly when the data is complex and deep, causing high-branch trees. Naive Bayes's tendency to assume feature independence can also hinder the overall accuracy since some features can have correlations and relationships. By incorporating Naive Bayes at certain decision tree leaf nodes, the model can reduce the complexity of the data, which can prevent overfitting.

Introduction

Decision Trees are effective at capturing feature interactions and splitting data into low-entropy subsets [3]. On the other hand, Naive Bayes, based on Bayes' theorem, is computationally efficient and well suited for high-dimensional data but relies on the assumption of feature independence[5]. However, Decision Trees often suffer from overfitting, particularly in complex datasets where deep trees memorize training data rather than generalizing well. Our work strikes a balance between these trade-offs by integrating Naive Bayes into Decision Trees, leveraging its probabilistic efficiency at intermediate and leaf nodes to prevent overfitting while maintaining a manageable computational load. This hybrid approach aims to improve both accuracy and scalability, making it a viable alternative for classification tasks involving structured data.

Related Work

Farid et al. conducted a similar study with the integration of decision trees and Naive Bayes classifiers [1]. The authors combined Naive Bayes with decision trees, and successfully handled large-scale datasets. It first used a Naive Bayes classifier to remove noisy data from the training set before making the decision tree. This varies from our approach, since it utilizes Naive Bayes beforehand in order to protect the training of the decision tree.

Renaningtyas et al. proposed a hybrid machine learning model combining the C4.5 Decision Tree algorithm and the Naive Bayes classifier to predict students' study periods and graduation predicates [6]. The Decision Tree model effectively handled the feature interaction, while the Naive Bayes, integrated at the leaf nodes, reduced complexity and enhanced accuracy. In contrast to this research, our approach suggests incorporating Naive Bayes not only at leaf nodes but also at various intermediate levels of the decision tree.

Dataset

Contraceptive Method Choice Dataset [2]

In order to assess the efficacy of our model, and what is the best configuration of our model, we chose to compare its accuracy against that of vanilla Naive-Bayes and Decision trees on several datasets, one of which being the Contraceptive Method Choice dataset. This dataset was acquired from the UCI Machine Learning Repository and is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey (Tjen-Sien Lim), tracking whether the type of contraceptive women in Indonesia use.

The dataset contains:

1. Wife's age	(numerical)	
2. Wife's education	(categorical)	1=low, 2, 3, 4=high
3. Husband's education	(categorical)	1=low, 2, 3, 4=high
4. Number of children ever born	(numerical)	
5. Wife's religion	(binary)	0=Non-Islam, 1=Islam
6. Wife's now working?	(binary)	0=Yes, 1=No
7. Husband's occupation	(categorical)	1, 2, 3, 4
8. Standard-of-living index	(categorical)	1=low, 2, 3, 4=high
9. Media exposure	(binary)	0=Good, 1=Not good
10. Contraceptive method used (class attribute)	(categorical)	1=No-use, 2=Long-term, 3=Short-term

Buys Computer Dataset

Buys Computer is a small, basic dataset that we used to test the methodology of our model. It is from a slideshow used in Dr. Yilmaz's ML 1 class.

The dataset contains:

1. Age	(categorical)	<=30, 31-40, >40
2. Income	(categorical)	low, medium, high
3. Student	(binary)	yes, no
4. Credit Rating	(binary)	fair, excellent
5. Buys_Computer	(binary)	yes, no

Seaborn Titanic Dataset [8]

The Seaborn Titanic dataset is a dataset containing information about the passengers on the Titanic, and the conditions they were in leading up to the crash. This dataset was readily available on SKLearn.datasets and preprocessed, allowing for an efficient and quick testing of the model. Most attributes are numerical, and there are three classes for classification. We selected this dataset in order to test on a highly-used, strongly distributed dataset.

1. Survival (class attribute)	(binary)	0 = No, 1 = Yes
2. Pclass (Ticket class)	(categorical)	1 = 1st, 2 = 2nd, 3 = 3rd
3. Sex	(binary)	Male, Female
4. Age	(numerical)	
5. Sibsp (# of siblings/spouses aboard)	(numerical)	
6. Parch (# of parents / children aboard)	(numerical)	
7. Ticket (ticket number)	(numerical)	
8. Fare (passenger fare)	(numerical)	

9. Cabin (cabin number) (numerical)
10. Embarked (port of Embarkation) C = Cherbourg, Q = Queenstown, S = Southampton

UCI Mushroom Dataset [4]

The UCI mushroom dataset is a categorical dataset that helps predict a species of mushroom. The dataset contains features about various mushrooms, and contains 8124 total samples. This dataset is highly correlated, with non-independent features and slight noise. This allows us to test our model on a more realistic dataset that contains noise and redundancies that might be found in a real-world application.

1. Poisonous (class attribute) (categorical) yes, no
2. Cap-shape (categorical) bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
3. Cap-surface (categorical) fibrous=f, grooves=g, scaly=y, smooth=s
4. Cap-color (binary) brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
5. Bruises (binary) bruises=t, no=f
6. Odor (categorical) almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
7. Gill-attachment (categorical) attached=a, descending=d, free=f, notched=n
8. Gill-spacing (categorical) close=c, crowded=w, distant=d
9. Gill-size (categorical) broad=b, narrow=n
10. Gill-color (categorical) black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y
11. Stalk-shape (categorical) enlarging=e, tapering=t
12. Stalk-root (categorical) bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?
13. Stalk-surface-above-ring (categorical) fibrous=f, scaly=y, silky=k, smooth=s
14. Stalk-surface-below-ring (categorical) fibrous=f, scaly=y, silky=k, smooth=s
15. stalk-color-above-ring (categorical) brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
16. stalk-color-below-ring (categorical) brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
17. Veil-type (binary) partial=p, universal=u
18. Veil-color (categorical) brown=n, orange=o, white=w, yellow=y
19. Ring-number (categorical) none=n, one=o, two=t
20. Ring-type (categorical) cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z

21. Spore-print-color (categorical) black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
22. Population (categorical) abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
23. Habitat (categorical) grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d

MNIST Data [7]

MNIST is a large database of small, square 28x28 pixel grayscale images of handwritten single digits between 0 and 9. It consists of a total of 70,000 handwritten images of digits, with the training set having 60,000 images and the test set having 10,000. All images are labeled with the respective digit that they represent. We chose to train on the MNIST dataset to examine how the model performs on datasets made up of mostly continuous data.

This dataset has 784 attributes, each value being numerical based on the grayscale values of each pixel of the individual images. The images were classified based on what digit they were thought to contain.

Methods

Our hybrid model, Baye-Tree, integrates Decision Trees and Naive Bayes to improve classification performance while reducing overfitting and computational inefficiency. The model begins as a standard, unpruned Decision Tree, where each node splits the data based on an attribute that maximizes information gain. However, the tree integrates Naive Bayes classifiers at designated depths and leaf nodes to make probabilistic predictions when appropriate.

Decision Tree Component

The algorithm selects a split that maximizes information gain, where information is calculates as:

$$I(x_1, x_2, \ldots, x_m) = -\sum_{i=1}^{m} p_i \log_2(p_i), \quad \text{where} \quad p_i = \frac{x_i}{|D|}$$

Naive Bayes Integration:

Naive Bayes is a probabilistic classifier that assumes conditional independence between features given the class label. The probability of a class B given feature vector A is computed using Bayes' theorem:

$$P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}$$

In Baye-Tree, we apply Naive Bayes at certain nodes based on a confidence threshold. If the classifier's probability is above this threshold, we use its prediction; otherwise, we continue with Decision Tree splits. This balances the bias-variance tradeoff, leveraging the efficiency of Naive Bayes while retaining the Decision Tree's hierarchical structure. Some of our key hyperparameters include max depth, a naive bayes confidence threshold, and a set of minimum samples for naive bayes calculation.

The code for this algorithm extends off of the decision tree codeThe pseudocode for this structure is as following:

Baye-Tree Algorithm Pseudocode

```
1: procedure HybridDecisionTree(dataset, target, maxDepth=3, minSamplesNB=10, nbThreshold=0.7, currentDepth=0, useNB=True, categoryRanges=None)
2:     if all target values in dataset are identical then
3:         return the common target value
4:     else if currentDepth $\geq$ maxDepth or only one feature remains then
5:         return most frequent target value
6:     end if
7:     if useNB and dataset size $\geq$ minSamplesNB then
8:         features $\leftarrow$ all features except target
9:         nbModel $\leftarrow$ create Naive Bayes classifier with categoryRanges
10:        train nbModel on features and target
11:        confidence $\leftarrow$ average maximum prediction probability
12:        if confidence $\geq$ nbThreshold then
13:            return Naive Bayes model with features and category ranges
14:        end if
15:    end if
16:    baseEntropy $\leftarrow$ calculate dataset entropy using target
17:    gainRatios $\leftarrow$ empty collection
18:    for each feature in dataset (except target) do
19:        valueEntropies $\leftarrow$ calculate entropy for each feature value
20:        valueProbs $\leftarrow$ calculate probability of each feature value
21:        weightedEntropy $\leftarrow$ sum of valueProbs $\times$ valueEntropies
22:        infoGain $\leftarrow$ baseEntropy - weightedEntropy
23:        splitInfo $\leftarrow$ calculate split information for feature
24:        if splitInfo $\neq 0$ then
25:            gainRatio $\leftarrow$ infoGain / splitInfo
26:        else
27:            gainRatio $\leftarrow 0$
28:        end if
29:        store gainRatio for feature
30:    end for
31:    bestFeature $\leftarrow$ feature with highest gainRatio
32:    tree $\leftarrow$ new tree node with bestFeature
33:    for each unique value of bestFeature do
34:        subset $\leftarrow$ dataset rows with this value, excluding bestFeature
35:        subtree $\leftarrow$ HybridDecisionTree(subset, target, maxDepth, minSamplesNB, nbThreshold, currentDepth + 1, useNB, categoryRanges)
36:        add subtree to tree under this value
37:    end for
38:    return tree
39: end procedure
```

Results & Discussion

The evaluation of the BaYe-Tree model was conducted using multiple datasets to assess its effectiveness in improving classification performance while reducing overfitting. The results varied based on dataset characteristics, particularly regarding feature independence and correlation.

Performance on the Buys Computer Dataset

- Naïve Bayes (NB) threshold: 0.8
- Minimum samples for NB calculation: 0.
- Result: The Naïve Bayes classifier was never invoked, and the model functioned as a standard decision tree.

Since Naïve Bayes was not called, it suggests that the dataset was more suited for a decision tree, possibly due to highly correlated features. The model did not improve classification accuracy, highlighting the necessity of careful threshold tuning.

Performance on the Contraceptive Method Choice (CMC) Dataset

- Naïve Bayes threshold: 0.4
- Decision Tree Accuracy: 47.46%
- Hybrid Model Accuracy: 53.90%
- Observation: The hybrid model consistently had a higher AUC, indicating better classification performance.

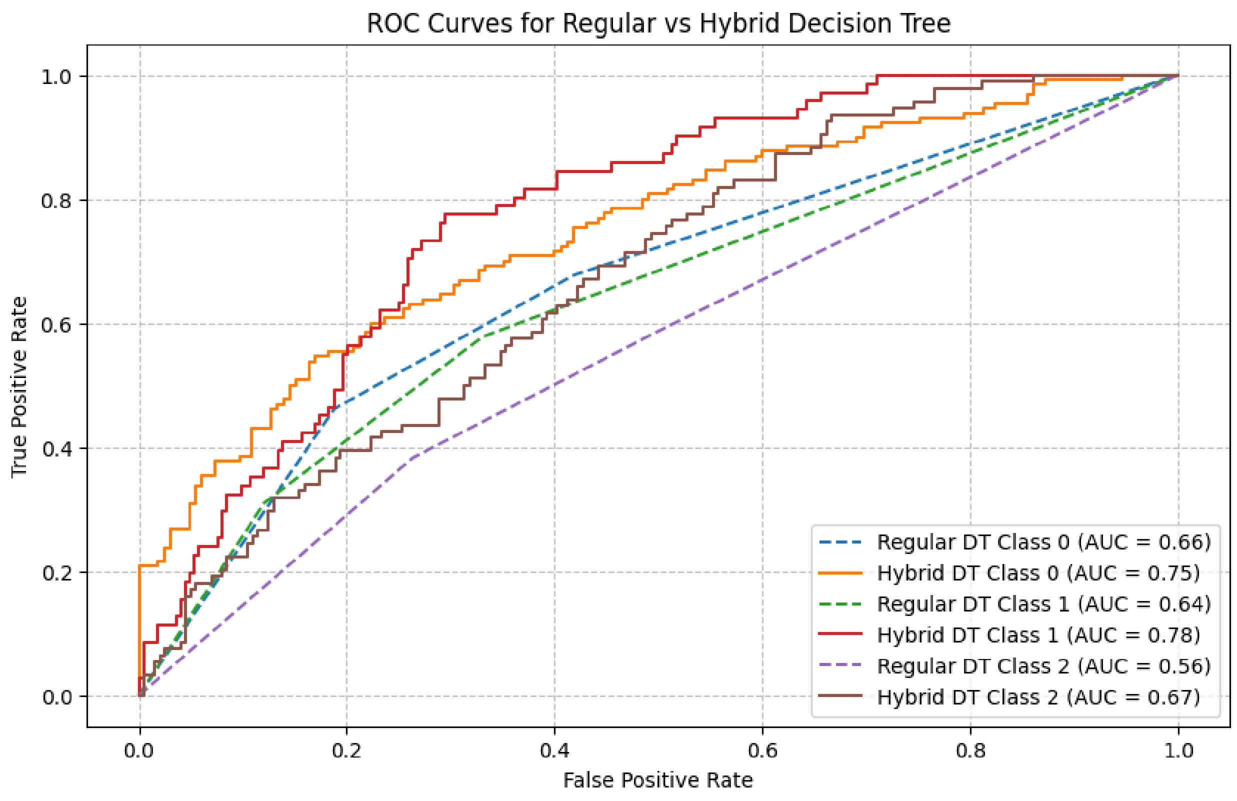

Figure 1: ROC Curves for Regular vs Hybrid Decision Tree on CMC Dataset

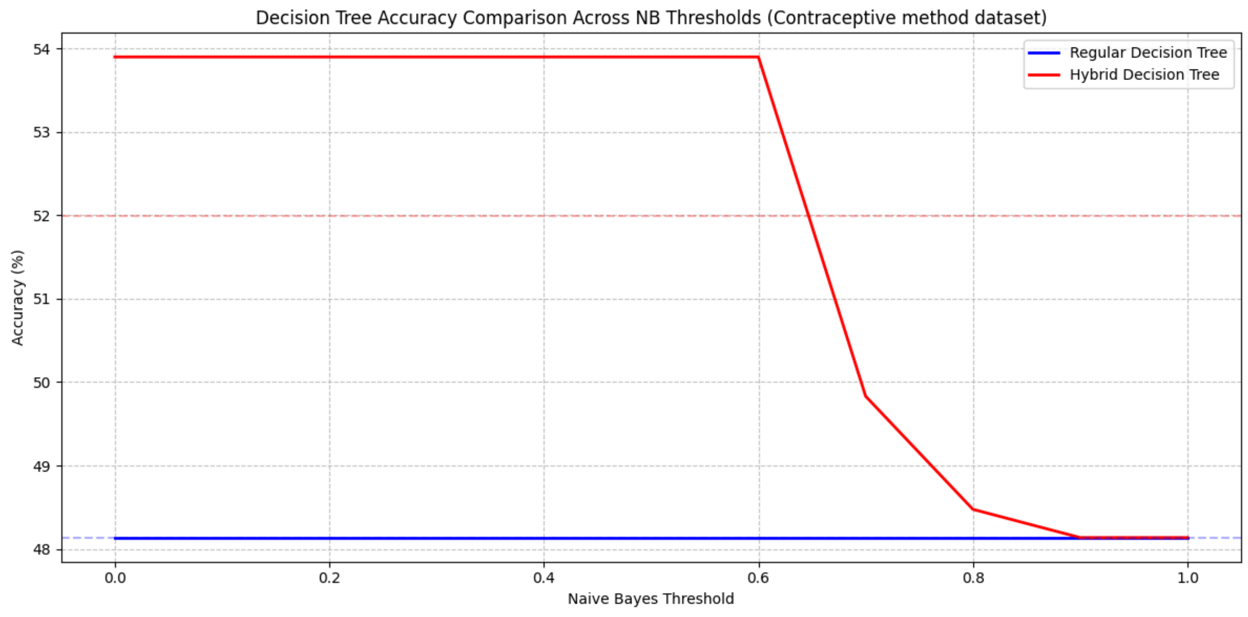

Figure 2: Accuracy comparison for Regular vs Hybrid Decision Tree on CMC Dataset

The integration of Naïve Bayes helped the model make better probabilistic decisions, leading to improved accuracy. This dataset likely benefited from a mix of feature independence and categorical data, which aligned well with Naïve Bayes' strengths.

Performance on the Titanic Dataset

- Naïve Bayes threshold: 0.3
- Decision Tree Accuracy: 81.46%
- Hybrid Model Accuracy: 80.90%
- Observation: The hybrid model slightly underperformed compared to the standalone decision tree.

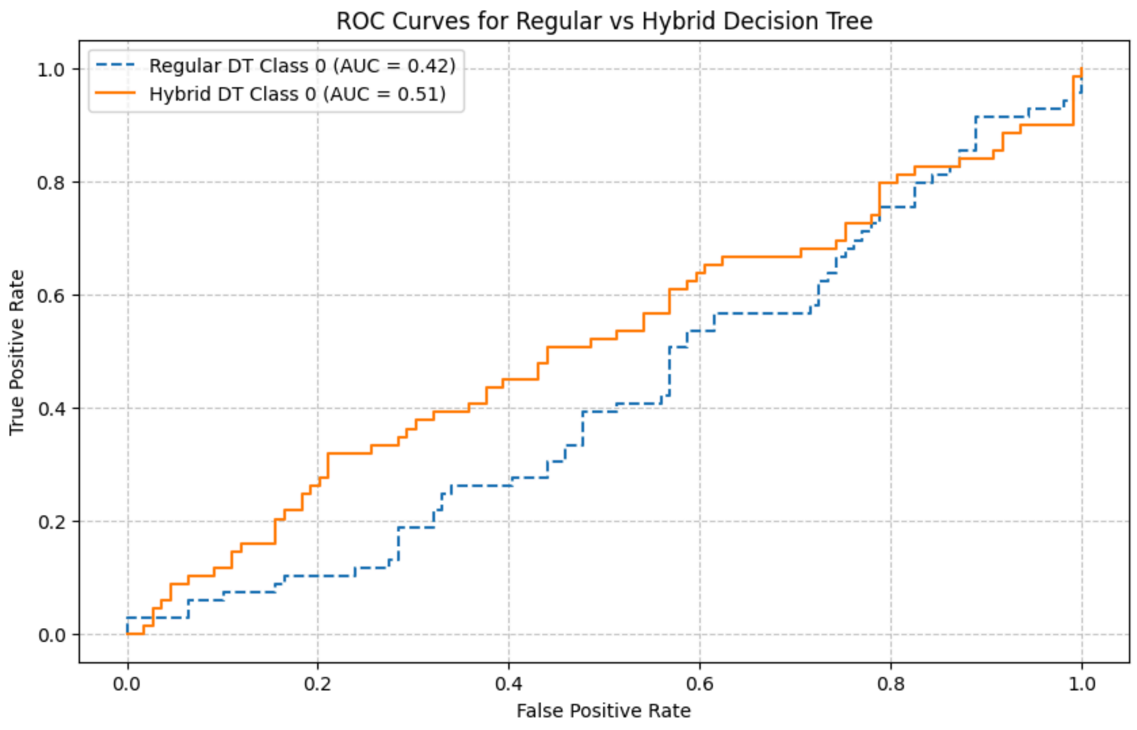

Figure 3: ROC Curve for Regular vs Hybrid Decision Tree on Titanic Dataset

While the decision tree maintained a marginally higher accuracy, the close results suggest that the hybrid approach did not significantly affect classification performance. One possible reason is that the dataset includes many dependent variables, which Naïve Bayes struggles with.

Performance on the UCI Mushroom Dataset

- Naïve Bayes threshold: 0.8
- Decision Tree Accuracy: 100.00%
- Hybrid Model Accuracy: 95.94%
- Observation: The hybrid model performed worse than the decision tree.

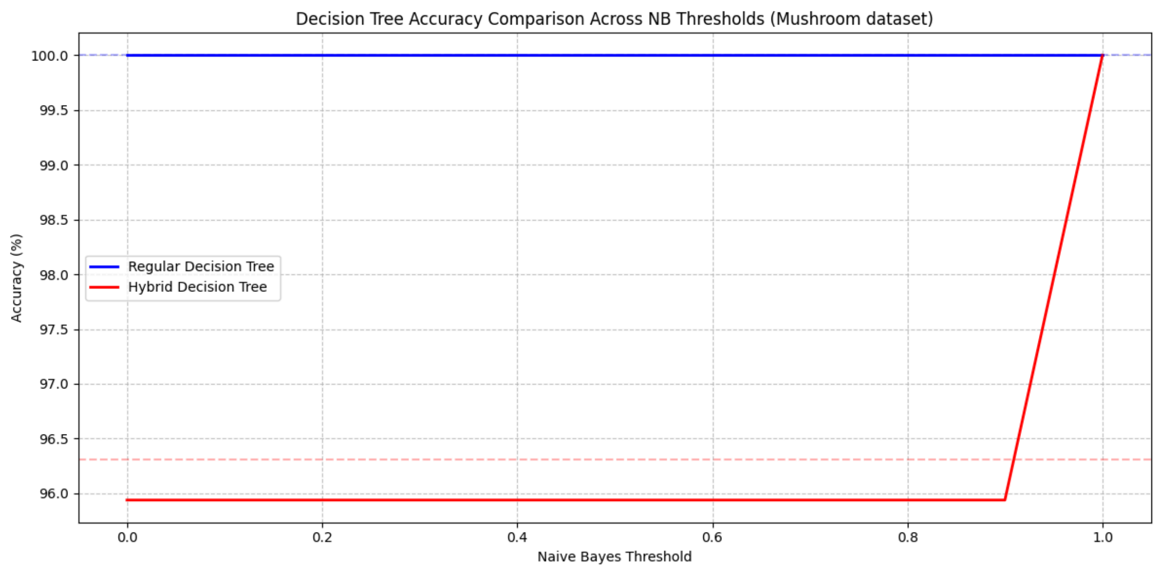

Figure 4: Hybrid vs Regular decision Tree Accuracy on Mushroom Dataset Across NB Thresholds

This dataset contained highly correlated features, which Naïve Bayes assumes to be independent. The decision tree was able to handle the noise and correlations more effectively, leading to a perfect classification score, while Naïve Bayes' overgeneralization reduced hybrid model accuracy. The effectiveness of the hybrid model is highly dependent on dataset characteristics. It performs well when features are independent but struggles with correlated data.The Naïve Bayes confidence threshold plays a crucial role in balancing the hybrid model's performance. Setting it too low or too high can either limit its influence or lead to overgeneralization. The stronger the independence between features, the better the hybrid model performs. In highly correlated datasets like the Mushroom dataset, the hybrid approach underperforms.

MNIST Dataset

In order to evaluate the Baye-tree's accuracy on a widely used benchmark for classification, this model was tested on the MNIST dataset, [7] which contains hand-written numbers with 784 pixels each. By incorporating MNIST, we aimed to assess whether our hybrid model could handle large feature spaces effectively and determine whether Naïve Bayes could contribute to digit classification, despite its assumption of feature independence. The hybrid decision tree received an accuracy of 83%, where our regular tree achieved 85% accuracy.

All Datasets

Dataset	Decision Tree Accuracy	Hybrid Tree Accuracy
Buys Computer	100%	100%
Contraceptive Method	48.14%	53.90%
Titanic Dataset	81.46%	80.90%
UCI Mushroom Dataset	100%	95.94%
MNIST Dataset	85%	83.7%

Table 1: Test Accuracies per Dataset

Conclusion/Future Work

The BaYe-Tree hybrid model aimed to integrate Decision Trees with Naïve Bayes to improve classification accuracy and reduce overfitting. Our results showed that the model performed well on datasets with independent features, such as the Contraceptive Method Choice (CMC) dataset, where it outperformed a standard Decision Tree by 6.44% in accuracy. However, it struggled on datasets with highly correlated features, such as the UCI Mushroom dataset, where the standalone Decision Tree achieved 100% accuracy while the hybrid model fell behind at 95.94% due to Naïve Bayes' assumption of feature independence. Given more time, future work would involve developing an adaptive Naïve Bayes threshold to dynamically adjust based on dataset characteristics and exploring feature transformation techniques like PCA to mitigate feature correlation issues. These improvements could further enhance the model's adaptability and efficiency across various datasets.

Contributions

Each team member contributed equally to the development and evaluation of the BaYe-Tree hybrid model. While Gus focused on researching and implementing the integration of Naïve Bayes with Decision Trees, Anmol handled dataset preprocessing, feature encoding, and model training. Lalit was responsible for analyzing results, optimizing hyperparameters, and compiling insights into the presentation. We split up the work equally amongst the team members to ensure a quality presentation and research paper.

References

[1] Farid, Dewan Md, et al. "Hybrid Decision Tree and Naïve Bayes Classifiers for Multi-class Classification Tasks." *Expert Systems with Applications*, vol. 41, no. 4, Mar. 2014, pp. 1937-46. *ScienceDirect*, https://doi.org/10.1016/j.eswa.2013.08.089.

[2] Lim, Tjen-Sien. "Contraceptive Method Choice." UCI Machine Learning Repository, 1999, https://doi.org/10.24432/C59W2D.

[3] Mienye, Ibomoiye Domor, and Nobert Jere. "A Survey of Decision Trees: Concepts, Algorithms, and Applications." *IEEE Access*, vol. 12, 2024, pp. 86716-27. *IEEE Explore*, https://doi.org/10.1109/access.2024.3416838. Accessed 9 Mar. 2025.

[4] "Mushroom." UCI Machine Learning Repository, 1981, https://doi.org/10.24432/C5959T.

[5] Peretz, Or, et al. "Naive Bayes Classifier – an Ensemble Procedure for Recall and Precision Enrichment." *Engineering Applications of Artificial Intelligence*, vol. 136, Oct. 2024, p. 108972. *ScienceDirect*, https://doi.org/10.1016/j.engappai.2024.108972. Accessed 9 Mar. 2025.

[6] Renaningtias, Nurul, et al. "Hybrid Decision Tree and Naïve Bayes Classifier for Predicting Study Period and Predicate of Student's Graduation." *International Journal of Computer Applications*, vol. 180, no. 49, 15 June 2018, pp. 28-34. *International Journal of Computer Applications*, https://doi.org/10.5120/ijca2018917329.

[7] "Recognizing Hand-written Digits." Scikit Learn, scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html. Accessed 21 Mar. 2025.

[8] Waskom, Michael. "Seaborn-data/Titanic.csv." *Github*, 21 Mar. 2014, github.com/mwaskom/seaborn-data/blob/master/titanic.csv. Accessed 9 Mar. 2025.