Assignment 2

July 17, 2024

1 Introduction

The `BigramDecisionTree` class is designed to train and predict words based on their bigrams using a Decision Tree Classifier from the scikit-learn library. This document provides a comprehensive overview of the approach, splitting criterion, stopping criteria, pruning strategies, and hyperparameters used in the implementation.

2 Approach

The approach involves the following steps:

1. **Extract Bigrams**: For each word, extract up to 5 unique bigrams.
2. **Create Feature Matrix**: Construct a feature matrix where each word is represented by the presence of its bigrams.
3. **Train the Model**: Fit a decision tree classifier on the feature matrix.
4. **Predict**: Use the trained classifier to predict words based on new bigram lists.

3 Splitting Criterion

The splitting criterion determines how the decision tree decides to split the data at each node. The criterion used in this implementation is:

- **Entropy**: This measures the impurity of a node, and the algorithm chooses splits that result in the lowest entropy. The formula for entropy is:

$$\text{Entropy} = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

  where p_i is the probability of class i at a particular node. Using entropy helps in creating a more balanced tree by focusing on informative splits.

4 Hyperparameters

The stopping criterion decides when to stop growing the decision tree:

- **Min Samples Split**: The minimum number of samples required to split an internal node is set to 2. This prevents the creation of splits that do not have enough data to justify them.
- **Min Samples Leaf**: Upon increasing the minimum samples leaf by one unit, the accuracy decreases by half. Therefore, we have set the minimum samples leaf to 1. This ensures that leaf nodes have enough data points, preventing overfitting to outliers.
- **Splitter**: The "best" splitter strategy is chosen to ensure the most optimal split at each node. This strategy evaluates all possible splits and selects the one that maximizes the criterion used (e.g., Gini impurity or information gain), leading to a more accurate and well-performing decision tree.

5 Result

Training Time: 4.7190332284000025 seconds

Memory Usage: 417122203.0 bytes

Accuracye: 97.50338687826592%

Testing Time: 0.9486196247999998 seconds

These metrics indicate that the model was trained relatively quickly, with efficient memory usage, and achieved high accuracy. The testing phase was also performed swiftly, ensuring the model's practical usability for predictions.