1.
○ Implement a decision tree and Naïve Bayes classifier for classification, with each of the above three ways of dealing with missing values. So you are experimenting with 6 scenarios.
○ Perform 5-fold cross validation and report precision, recall, and F1-scores for each of the 6 scenarios

Introduction

Based on a data collection comprising 16 variables and a class label for each representative, this study seeks to examine the voting patterns of the US House of Representatives in 1984. This project's key research question is: To what extent can we infer a representative's party allegiance from their voting history? This objective is important because it allows us to asssess the effectiveness and restrictions of various categorization systems and pinpoint the features that have the greatest bearing on party affiliation prediction.

The UCI Machine Learning Repository, a well-known source of data sets for machine learning research and applications, provided the data set that I used for this project. 435 instances and 17 attributes, comprising 16 features and one class label, make up the data set. The class label designates whether a representative is a member of the democratic or republican parties, and the features represent each representative's votes on 16 important issues in 1984.

Methods

The data set contained some missing values for some of the features, therefore I either discarded those occurrences where the feature values were missing, treated "missing" as if it were a value, or imputed using the value that occurred the most frequently for each feature.

I utilized decision trees and Nave Bayes as classification methods to analyze the data, which are supervised learning methods that try to provide a class label to an instance based on its features.

Results

The main results of this analysis are shown in the following:
Discard instances that have missing feature values:
Decision tree Precision: 0.96
Decision tree Recall: 0.94
Decision tree F1-score: 0.95
Naïve Bayes Precision: 0.95
Naïve Bayes Recall: 0.96
Naïve Bayes F1-score: 0.96

Treat “missing” as if it is a value (and thus a binary feature becomes a ternary, or three-valued, feature:
Decision tree Precision: 0.96
Decision tree Recall: 0.97
Decision tree F1-score: 0.96
Naïve Bayes Precision: 0.95
Naïve Bayes Recall: 0.95
Naïve Bayes F1-score: 0.95

Impute missing values:
Decision tree Precision: 0.96
Decision tree Recall: 0.94
Decision tree F1-score: 0.95
Naïve Bayes Precision: 0.95
Naïve Bayes Recall: 0.93
Naïve Bayes F1-score: 0.94

The results show that we can predict the party affiliation of a representative based on their voting record with a high degree of accuracy using different classification techniques.
The decision tree classifier achieved the highest precision of 96% among all the models that we tested, followed by the Naïve Bayes classifier with 95% precision.

2.
For what type of dataset would you choose decision trees as a classifier over Naive Bayes?

Compared to Naive Bayes, decision trees are better able to handle missing values.
Decision trees work better with datasets that are non-linear or have intricate feature relationships.
Compared to Naive Bayes, decision trees are easier to understand and visualize, which might aid in deriving the justification for the predictions.

For what type of dataset would you choose Naive Bayes as a classifier over decision trees?

Naive Bayes are better suited for datasets with independent, linear features.
For high-dimensional datasets, naive Bayes performs more effectively than decision trees since it uses less memory and computing resources.
Naive Bayes are more resistant to noise and outliers than decision trees because they use probabilistic models.