Algorithm implementation:

Decision Tree(ID3)

Overview

The project aims to develop a custom implementation of the decision tree algorithm in Python and then test the model's performance on several datasets.

Implementation details

Github repository: https://github.com/kiiril/ID3

The repository contains the following files:

- *decision_tree.py* - actual algorithm implementation
- *examples.ipynb* - few examples of applying the algorithm on toy datasets

The ID3(Iterative Dichotomiser 3) algorithm was selected for this project since it is particularly well-suited for classification tasks. The main principle is to recursively partition the data into subsets based on the feature that provides the highest information gain at each step and build the tree that is used in class(label) prediction. The algorithm can handle categorical features as well as numeric ones.

Here are the crucial methods that contribute to the construction and functionality of the decision tree:

- *entropy()* - calculates and returns the entropy of feature or class
- *info_gain()* - checks for the type of the feature and calls the appropriate method
- *info_gain_continuous()* - calculates and returns the information gain and the best splitting value for the categorical feature
- *info_gain_discrete()* - calculates and returns the information gain for the discrete feature
- *split()* - splits data based on the condition (feature with the largest info gain)
- *_fit()* - an auxiliary function to recursively build the tree
- *predict_one()* - traverses the tree and predicts the label for one instance
- *predict()* - predict labels for all features

To evaluate the model's performance, the Iris and Breast Cancer datasets were used. The algorithm achieved an average accuracy of **93.3%** on the Iris dataset and **94.7%** on the Breast Cancer dataset, indicating strong performance across both cases.

Tools: Python, NumPy (simplify work with arrays), sklearn (prebuilt functions for evaluation)