Implementation of an algorithm to learn decision trees.

# Introduction

What the problem is what I am doing why interesting and what my contribution is

# Background

#### **Binary Classification**

Explain what binary classification is, focusing on its importance in machine learning. Describe how binary classification works to categorize instances into one of two classes based on input features. Use examples to illustrate its application in real-world scenarios.

#### **Decision Trees**

Introduce decision trees as a method used in binary classification. Use notations to describe their structure, including nodes, edges, leaves, and decision points. Explain how these components work together to make classifications.

##### Learning Decision Trees

Detail the process of constructing decision trees from a given dataset. Define the computational problem of finding the smallest possible decision tree and explain why minimizing tree size is beneficial, such as reducing overfitting and improving interpretability.

#### **Known Algorithms**

Discuss two to three known algorithms for constructing decision trees. Categorize these algorithms into heuristic and exact types:

* **Heuristic Algorithms**: Describe common heuristic approaches like the ID3, C4.5, or CART algorithms, and discuss their prevalence in industry due to their efficiency and practical results.
* **Exact Algorithms**: Explain algorithms designed to find the smallest possible decision tree, emphasizing their computational expense and how they guarantee finding the optimal solution.

#### **Algorithm to Implement**

Choose one algorithm that you will implement for your project. This could be either a heuristic or an exact algorithm, depending on your project's focus and the complexity you are ready to handle.

##### How the Algorithm Works

* **Pseudo Code**: Provide pseudo code for the algorithm, explaining each step in detail.
* **Running Time**: Analyze the running time of the algorithm, discussing its efficiency and any significant factors that affect performance.

##### Complexity and Parameterization

Discuss the complexity of the algorithm, focusing on parameterized complexity. Explain what parameterized complexity is and how it applies to your chosen algorithm. This may include defining parameters like tree depth or the number of features.

##### Correctness of the Algorithm

Validate the correctness of the algorithm. Explain how you can ensure that the algorithm correctly builds a decision tree that accurately classifies the input instances. This might involve discussing proof strategies or empirical validation methods.

### Conclusion

Conclude the "Background" section by summarizing the key points discussed and linking them to the subsequent sections of your report, where you will detail your implementation and the results of your experiments.

This structure will help to clearly lay out the foundational concepts and methods your project builds upon, while also positioning your own work within the broader field of machine learning and decision tree learning.

### 1. Introduction

* 1.1. The Problem
* 1.2. Objectives of the Study
* 1.3. Significance of the Study
* 1.4. Contributions of This Work

### 2. Background

#### **2.1. Binary Classification**

* 2.1.1. Definition and Overview
* 2.1.2. Applications in Industry

#### **2.2. Decision Trees**

* 2.2.1. Structure and Components
* 2.2.2. Notational Description

##### 2.3. Learning Decision Trees

* 2.3.1. The Process of Building Decision Trees
* 2.3.2. Computational Problem: Finding the Smallest Decision Tree
* 2.3.3. Importance of Smallest Decision Trees

#### **2.4. Overview of Existing Algorithms**

* 2.4.1. Heuristic Algorithms
  + 2.4.1.1. Common Heuristic Approaches
  + 2.4.1.2. Advantages and Use in Industry
* 2.4.2. Exact Algorithms
  + 2.4.2.1. Definition and Examples
  + 2.4.2.2. Computational Expense and Optimization

### 3. The Selected Algorithm for Implementation

#### **3.1. Description of the Algorithm**

* 3.1.1. Choice of Algorithm
* 3.1.2. Suitability and Relevance to the Project

#### **3.2. Implementation Details**

* 3.2.1. Pseudo Code
* 3.2.2. Analysis of Running Time

#### **3.3. Complexity and Parameterization**

* 3.3.1. Complexity Analysis
* 3.3.2. Parameterized Complexity Explained

#### **3.4. Correctness of the Algorithm**

* 3.4.1. Proving Correctness
* 3.4.2. Validation Methods

### 2.1. Binary Classification

#### **2.1.1. Definition and Overview**

Binary classification is a type of supervised learning algorithm where the task is to predict one of two possible categories, or classes, based on input features. Each instance in binary classification is labeled with one of these two classes, making it a clear choice for problems where dichotomous outcomes are involved (James et al., 2013). In contrast, multiclass classification involves categorizing instances into three or more classes, where the decision-making process is inherently more complex due to the increased number of outcomes (Hastie et al., 2009).

#### **2.1.2. Applications in Industry**

Binary classification has widespread applications across various industries. For instance, in healthcare, binary classifiers determine patient diagnoses as either 'positive' or 'negative' for specific conditions, which is crucial for early intervention (Kourou et al., 2015). In the realm of finance, these algorithms are employed to distinguish fraudulent from legitimate transactions, thereby enhancing security measures (Bhattacharyya et al., 2011). Furthermore, multiclass classification is utilized in retail to categorize customer feedback into multiple sentiment classes, helping businesses tailor marketing strategies and improve customer service (Aggarwal and Zhai, 2012). Both types of classification play vital roles in automating and optimizing decision-making processes across different sectors.

Binary and Multiclass Classification

Definition and Overview

Binary classification is supervised learning algorithm where the goal is to predict, given input features, one of two potential classes or categories to which an instance belongs. Each instance in binary classification is labelled with one of the two classes (Lorena, Carvalho, and Gamma, 2008), therefore it is an obvious choice for problems with opposite outcomes such as “yes” or “no”, “positive” or “negative”, “healthy” or “diseased”.

In contrast, multiclass classification entails classifying instances into three or more classes (Lorena, Carvalho, and Gamma, 2008). This scenario is common in situations where outcomes are not limited to two possibilities, such as classifying types of crops, recognising various languages or diagnosing multiple types of diseases. Due to the increased number of possible outcomes, the decision-making process is inherently more complex.

Classification Instances

A classification instance in machine learning refers to a dataset made up of multiple data entries that are processed by a classification algorithm. Each of these examples, which are part of a larger dataset, consist of a label and set of features. The features are the measurable properties or characteristics of the instance, whereas the label designates the category to which the instance is assigned based on the features.

Binary Classification instance

In binary classification each example is labelled with one or two possible classes. Here the classification instance can be defined as a tuple C = (E, F, T), where:

* + E
  + F
  + T

This definition is taken from …

FORMAL DEFINITION

Multiclass Classification Instance

Multiclass classification extends the concept of binary classification by involving three or more classes into which the examples can be classified. Here a classification instance is still defined as a tuple C = (E, F, T), but with an increased domain for the classification function:

* + E
  + F
  + T

Both classification instances can be represented as a table, where each column could represent a feature and the final column could represent the class of the example.

Application in industry

Binary Classification has widespread app

Decision trees

Decision trees represent a hierarchical structure for decision-making, where each node signifies a decision based on certain attributes, and the branches denote the outcome of these decisions (Kingsford & Salzberg, 2008) The process initiates from the root node and progresses through internal nodes (representing attributes tests) to the leaf nodes, which hold the decision outcomes – class labels for classification tasks or continuous values for regression tasks. This process of systematically dividing the entire dataset into smaller subsets mirrors the divide and conquer approach often used in problem solving. By applying this method decision trees aim to simplify complex dataset into subsets that are easier to analyse and make predictions from. (Kingsford & Salzberg, 2008)

The simplicity and interpretability of decision trees make them suitable for a wide array of applications:

Classification and prediction: Their primary use in categorising instances into distinct classes based on attribute values supports applications in a variety of fields such as medical diagnosis, weather forecasting and sales predictions among others ()

Regression: Beyond classification, decision trees can be predictors of continuous outcomes, making them invaluable tools in forecasting sales, evaluating real estate prices, and other quantitative analyses (reference)

The representation of decision trees is both straightforward and visually engaging, enhancing their appeal:

Nodes: The nodes in the decision tree are the entities that make decisions. The root node represents the entire dataset, internal nodes correspond to attribute tests which split the dataset, and leaf nodes represent the outcome of these decision paths.

Branches: These represent the decision outcome at each node, guiding the path later nodes or leaf outcomes

This structured approach allows decision trees to transparently communicate the logic behind decision making processes, which makes it easier to grasp and apply them in a variety of disciplines (reference)

DIAGRAM EXAMPLE

Binary Classification Instance

Reference

Lorena, A., Carvalho, A., & Gama, J., 2008. A review on the combination of binary classifiers in multiclass problems. Artificial Intelligence Review, 30, pp. 19-37. <https://doi.org/10.1007/s10462-009-9114-9>.

Kingsford, C., Salzberg, S. What are decision trees?. *Nat Biotechnol* **26**, 1011–1013 (2008).

Talk about:

* Algorithms which are known 2/3, show my understanding, link to my work
* 2 types, heuristic, exact
* Show smallest decision trees are needed

Explain:

* Algorithm to implement.
* How algorithm works with pseudo code and running time
* Complexity – parameterised , what is this
* Completeness of the algorithm

## Method

* Design of solution
* How I implemented, project management methodology
* What improvements I did
  + Extended domain to discrete
  + Removed symmetries.
  + Error that caused loop
* Challenges of implementation
* Evidence of good coding practice and version control
* Detailed description of software implementation with justifications

# Validation and evaluation

* Run it on different benchmarks tests
  + How long it takes.
  + Compare to SAT algorithm
* to find what s is
  + From 1 and go up to find solution
* Extended domain to discrete, performance of this
* How I showed that its correct
* Improvements, proving they don’t affect the solution
* Testing structure, how we test the correctness

Size

Conclusion

* Ideas for future work

Problem definition

* Goal is to decision tree with smallest mumb

# Self-appraisal:

* Considered how my work could be extended?
* Included mature personal reflection which leads to lessons learnt?
* Suggested how the problems encountered might be avoided?
* Considered legal, social, ethical and professional issues, with justification if one or more are not relevant?

Remove symmetries.

Enumerate everything to avoid do same thing again.

Do some kind of ordering.

Avoid repeating features in branch

Getting same trees in different ways/orders

Have an ordering on the feature

* **Don’t consider putting a feature that is smaller in the ordering above(higher in tree) a feature that is larger, only on the path you are extending,**
* **In report show we still generate all possibilities but we don’t generate the same tree twice**

**For the error, when inserting feature, make sure everything below is reachable before adding, else skip**