Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrics and Operations Research

Medical diagnosis using Imbalanced Decision Trees as Integer Programs with out-of-sample analysis

Maurice de Kort (564986)

Supervisor:	Dr. R.M. Badenbroek
Second assessor:	Dr. M.H. Akyuz
Date final version:	2nd July 2023

Abstract

The application of machine learning algorithms in medical diagnosis has gained significant attention due to their potential to enhance accuracy, efficiency, and objectivity in the decision-making process. Decision tree models in particular are interesting for this purpose, due to their interpretability and simplicity. In this paper, we investigate the performance of Decision Trees as Integer Programs (DTIP), which is a way of encoding the learning of decision trees of a fixed depth as an integer optimization problem. We test this method and a variation of it with an extra penalty on false positive predictions on two medical diagnosis datasets. We find that DTIP with an existing decision tree as a starting solution always finds trees that outperform decision trees given by the greedy heuristic CART, both in-sample and out-of-sample. Furthermore, we show that imbalanced DTIP can be applied to limit the number of false positive predictions and can for some imbalanced datasets even improve the out-of-sample accuracy.

1 Introduction

Advances in machine learning have revolutionized numerous fields, and one area where its potential is increasingly being recognized is in medical diagnosis. The ability of machine learning algorithms to analyze vast amounts of complex data, detect patterns, and make accurate predictions holds great promise for enhancing diagnostic capabilities. Traditionally, medical diagnosis has relied heavily on the expertise of healthcare professionals and their interpretation of clinical symptoms, laboratory tests, and medical imaging. However, this process can be subject to human error and limitations in analyzing large and heterogeneous datasets. Machine learning offers the possibility to overcome these issues. A machine learning tool that is especially interesting in the realm of medical diagnosis are decision tree methods. These methods create insightful models, that are easy to understand and apply, and have proven to be very effective for diagnostic purposes (Podgorelec et al. (2002)). In this paper, we specifically focus on Decision Trees as Integer Programs (Verwer and Zhang (2017)) to create decision trees that outperform those created by more traditional methods, such as Classification and Regression Trees (Breiman et al. (1984)). We test our methods on two medical datasets: one with data of breast cancer tumours, and the other one with data of potential heart disease patients.

Breast cancer is one of the most common forms of cancer among women. When a tumor is identified, it is important to examine whether it is benign or malignant. Benign tumors can be removed and do not reappear. Malignant tumors grow much faster than benign tumors and can be fatal when left untreated. Treatment of breast cancer can be highly effective, especially if it is identified early. For this reason, classifying breast cells as benign or malignant is a crucial task, and machine learning techniques could help a lot in identifying cancerous cells and thereby increase the survival rate of breast cancer. In this paper we will specifically focus on optimal decision trees to classify a tumor as benign (B) or malignant (M) based on a set of features that are computed from digitized images of fine needle aspirates (FNA) of breast masses, derived by Street et al. (1993).

Heart diseases, or cardiovascular diseases (CVDs) in general, are the leading cause of death globally, taking an estimated 17.9 million lives each year. As stated by the World Health Organization (WHO): "Identifying those at highest risk of CVDs and ensuring they receive appropriate

treatment can prevent premature deaths". This shows the importance of proper diagnosis. In this paper we use optimal decision trees to identify whether or not a patient suffers from any kind of heart disease based on a number of features collected by Janosi et al. (1988).

Decision trees are a popular tool for classification. They provide a simple scheme of splitting rules that allow the user to classify new instances extremely efficiently. Moreover, the outcome is highly interpretable, unlike many other machine learning methods. This makes it especially suitable for medical purposes, because the doctors who might use the outcomes to identify breast cancer, will likely not have a lot of knowledge on statistics, calling for a model that is easy to understand for everybody.

Typically, decision trees are constructed following the approach proposed by Breiman et al. (1984): Classification and Regression trees, abbreviated as CART. This heuristic starts at the root of the tree, and works downwards, recursively determining the best split in every node. The downside of this greedy approach is that in each split the effect on future splits is not taken into account, resulting in sub-optimal decision trees. This issue is addressed by Verwer and Zhang (2017), who propose a way of programming the problem of learning optimal decision trees as an integer optimization problem (DTIP). Using this formulation, we can apply powerful MIP-solvers to find optimized trees. Although this method requires more running time, it has been shown to improve performance over CART, when provided with a decision tree found by CART as a warm start. Moreover, their proposed formulation allows for creating decision trees with non-standard objectives.

We utilize the flexibility of the methods introduced by Verwer and Zhang (2017) by including a constraint for counting false positives and adding it to the objective function with a multiplier λ. By increasing this value λ, we can reduce the amount of false positive classifications made by the decision tree. This is a useful feature, because in practice we might not want to wrongfully diagnose a patient as having a malignant type of breast cancer or having a heart disease. Furthermore, the breast cancer dataset is quite imbalanced, as only 37% of the patients suffer from a malignant tumor and 63% have a benign tumor. By including this penalty term in the objective of DTIP, this method is able to handle this imbalance better than heuristic-based decision trees. We refer to this method as imbalanced DTIP.

The idea of decision trees has been around for a long time, but since constructing optimal classification trees is known to be NP-hard (Hyafil and Rivest (1976)), earlier research focused on greedy heuristics. Some of the most popular heuristics are Classification and Regression Trees (CART), introduced by Breiman et al. (1984), which choose the optimal split based on the Gini impurity, and ID3, introduced by Quinlan (1986), which uses entropy-based information gain to determine the optimal splits in each layer.

Recently, there has been more focus on building optimal classification and regression trees. Similar ideas have been explored in the past, for example in Bennett and Blue (1996), but optimal classification trees only started to flourish with the recent advancements in hardware. These methods have now become feasible due to the enormous increase in computational speed by optimization solvers (Bixby (2012)).

Bertsimas and Dunn (2017) present the problem of creating an optimal decision tree as a mixed-integer optimization problem. Their formulation can be extended to create multivariate decision trees, which can split on multiple features in each node. Another formulation of the problem was introduced by Verwer and Zhang (2017). They propose an integer formulation that can be adjusted to make discrimination-free decision trees and improve learning from imbalanced data. Both find that optimal decision trees systematically outperform greedy heuristics, such as CART.

A downside to optimal decision trees is the computational burden. To speed up the algorithm, Verwer and Zhang (2019) propose a binary formulation that removes the dependency on the dataset size, thereby improving the scalability of optimal classification trees. Some other works that introduce methods to improve scalability are Hu et al. (2020), Lin et al. (2020) and Demirović et al. (2020). Lastly, Aghaei et al. (2022) propose a flow-based formulation that has a provably stronger linear relaxation than preceding methods.

Regarding the breast cancer data, the first analysis was done by Street et al. (1993), who also derived the data. With a linear-programming-based inductive classifier, they were able to obtain a 10-fold cross-validation accuracy of 97%. Although these results are great for prediction, they are not very useful when trying to understand the data, because the applied model is difficult to interpret. A more recent study of this dataset was done by Mohammad et al. (2022), using several Machine Learning techniques for classification and clustering. One of the methods they tested was a J48 decision tree, which had an accuracy of 93%. We expect to see that the DTIP achieves a higher accuracy and is more capable of handling the imbalanced nature of the data.

The dataset for heart diseases was obtained by Janosi et al. (1988). Some related works that discuss the effectiveness of different types of decision tree based classification methods are Elyan and Gaber (2016), who applied class decomposition to improve the performance of Random Forests, and Luna et al. (2017), who proposed Tree-Structured Boosting (TSB) to create decision trees that outperform CART. Furthermore, it is known that on this dataset Random Forest classification has an accuracy of 80.26%, which we can consider a baseline for our methods (Dua and Graff (2017)).

The main question in our research is: How well does DTIP in the context of medical diagnosis? To answer this, we investigate the following sub-questions:

1. What is the rate of correctly classified patients when using regular DTIP?
2. What is the out-of-sample performance of DTIP?
3. Can we reduce the number of out-of-sample false positive predictions with imbalanced DTIP?

The outcomes of our research are relevant for the medical world, not only for detection of breast cancer and heart diseases, but for the detection of diseases in general, as this method can be applied in many different context. Furthermore, our research also contributes to the field of classification trees, as to the best of our knowledge, no literature exists on the out-of-sample

performance of DTIP.

Verwer and Zhang (2017) show that DTIP with a CART solution as a warm start can be used to create decision trees that always achieve an equally high, or higher rate of correctly classified patients within the training dataset than the decision tree provided by CART. Our results show that this also holds within the context of medical diagnosis. Moreover, using DTIP with a warm start on medical diagnosis datasets, we find decision trees that achieve a higher out-of-sample accuracy than those produced by CART. This shows that DTIP does not simply overfit to the training dataset, but gives a model that is closer to the true structure of the data. Furthermore, we find that we can use imbalanced DTIP to reduce the number of false positive out-of-sample predictions that are made by the resulting decision tree model and even increase the out-of-sample accuracy of the tree for imbalanced datasets. However, we find that a higher penalty does not always lead to a reduction in the average number of false positive out-of-sample predictions, and therefore parameter tuning is required to find the penalty that yields the lowest number of false positive out-of-sample predictions.

Our paper will be structured as follows: Section 2 discusses the datasets that we use and describes the data transformation performed. Then, in Section 3, we give our formulation of DTIP and imbalanced DTIP, and we outline our procedure for the experiments. Next, Section 4 discusses the outcomes of our experiments and lastly, Section 5 summarizes our results and conclusions and discusses the limitations and options for further research.

2 Data

2.1 Description of the data

The datasets used in this paper are obtained from the UCI Machine Learning Repository (Dua and Graff (2017)). We construct optimal decision trees for the same datasets as Verwer and Zhang (2017): "Iris", "Pima Indian Diabetes" and "Bank Marketing". We only use the datasets for classification trees, because our paper does not focus on regression trees. The "Iris" dataset is a small dataset, consisting of 150 datapoints with 4 attributes and 3 different classes. The "Pima Indian Diabetes" dataset (Kahn (1994)) contains 768 datapoints with 8 attributes and 2 classes. The largest dataset is the "Bank Marketing" dataset (Moro et al. (2012)), which consists of 4521 datapoints with 16 attributes. After converting each of the categorical features into binary variables for each category, we end up with 48 attributes. For the breast cancer diagnosis, we will use the "Breast Cancer Wisconsin (Diagnostic)" dataset. This dataset contains 569 datapoints with 30 attributes and two classifications: Benign (B) or Malignant (M). An important aspect of this dataset is the imbalance, as only 37% of the patients (212 datapoints) suffer from a malignant tumor. This could lead to skewed decision trees when applying regular decision tree methods, and therefore promotes the use of imbalanced DTIP.

For the heart disease diagnosis we use the "Heart Disease" dataset. This dataset consists of 303 datapoints and 13 attributes. We remove all datapoints with missing values, such that we end up with 297 datapoints. We attempt to predict the presence of a heart disease, indicated

by values 1, 2, 3 and 4, which we collapse to a value of 1. If the patient does not have a heart disease, this is indicated by a value of 0. With this definition, 46% of the patients are classified as 1 (137 datapoints) and 54% are classified as 0 (160 datapoints), making this dataset less imbalanced than the breast cancer dataset.

2.2 Data transformation

We transformed the data using the following procedure for each column: First, we sort all of the feature values in increasing order. We merge the sequential feature values that all belong to datapoints of the same class into one value. Then we count how many times each feature value occurs and we map the most frequently occurring feature value to 0. Finally, we map all other feature values to integer values around 0, while retaining the original ordering of the feature values. With this procedure, we try to mimic the data transformation performed by Verwer and Zhang (2017) to the best extent. Although our transformation follows their description, we see some differences in the outputted values compared to those provided in their paper. This could potentially lead to minor differences in computation times.

To make it more clear how this procedure works in practice, we provide a small example. We consider a set of 7 datapoints and 2 classifications, as displayed in the table on the left in Figure 1. We start by sorting the data. After this, we merge the feature values that occur multiple times (7.3 in this example) and we also merge sequential feature values that have the same class (6.2 and 6.5). We cannot merge 5.1 with 6.2 and 6.5, because it belongs to a different class. Furthermore, we cannot merge 7.3 with any other feature value, because not all datapoints with value 7.3 belong to the same class. After we made this merge, we end up with 4 (sets of) feature values, and we see that the value 7.3 occurs the most, meaning that it will be assigned a value of 0. The other feature values are then mapped to values around 0, such that we end up in the table on the right.

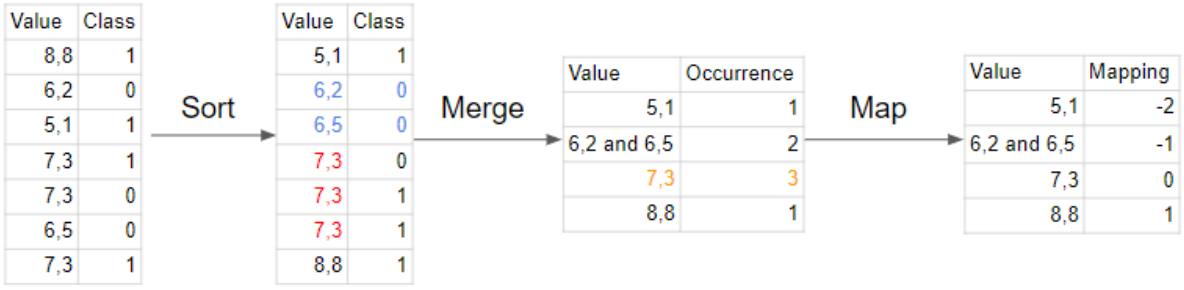

Figure 1: Example of data transformation

3 Methodology

Variable	Type	Definition
n	Constant	Number of rows (observations) in data
m	Constant	Number of columns (features) in data
k	Constant	Depth of the tree
u	Constant	Number of nodes in tree (excluding leaves)
v	Constant	Number of leaves in tree
$d(j)$	Constant	Depth of node j in tree (root has depth 0)
$v(r,i)$	Constant	Feature value of row r, feature i
$t(r)$	Constant	Target value (class) of instance r
LF	Constant	Minimum value over all features
UF	Constant	Maximum value over all features
T	Set	Set of different target values
f_{ij}	Binary decision variable	1 if feature i is used in decision rule of node j, 0 else
c_j	Integer decision variable	Threshold in decision rule of node j
l_{hr}	Binary decision variable	1 if path of data row r goes left at depth h, 0 else
p_{lt}	Binary decision variable	1 if prediction in leaf l is target (class) t, 0 else
e_r	Binary decision variable	Prediction error for data row r

Table 1: Summary of the notation used in the formulation of DTIP

3.1 DTIP formulation

Our formulation of Decision Trees as Integer Programs (DTIP) is largely based on the formulation introduced by Verwer and Zhang (2017), with some changes in notation. The notation that we use is summarized in Table 1. Note that u and v are constant once we know k, because we always construct a complete tree. Since the depth of the tree is fixed, we can compute u as $2^k - 1$ and v as 2^k. The depth of the tree in this formulation is defined as the number of edges between the root of the tree and a leaf. Also note that for the breast cancer data, we have only two possible classifications, benign (B) and malignant (M), so $t(r) \in \{\text{B}, \text{M}\}$ for any row r. Similarly, for the heart disease data we have $t(r) \in \{0, 1\}$, where 0 means the patient has no heart disease, and 1 means they do. Furthermore, we define $M_L(r)$ and $M_R(r)$ for row r as follows:

$$M_L(r) = \max\{v(r,i) - LF | i = 1, ..., m\}$$

$$M_R(r) = \max\{UF - v(r,i) | i = 1, ..., m\}$$

$M_L(r)$ is defined as the maximum deviation from the lower bound over all features values of a specific row. This corresponds to the highest possible deviation from the threshold if row r goes left at a certain node and is therefore used for a tight big-M formulation. Likewise, $M_R(r)$ corresponds to the highest possible deviation from the threshold if row r goes right at this node. Finally, $\text{dlr}(h, j, r)$ is a binary function that yields 1 when row r follows the path to node j at depth h, and 0 otherwise. It can be derived from the variable l_{hr} in the following way:

$$\text{dlr}(h, j, r) = \begin{cases} l_{hr} & \text{if path to node/leaf } j \text{ goes left at depth } h \\ 1 - l_{hr} & \text{if path to node/leaf } j \text{ goes right at depth } h \end{cases}$$

Using this notation, we can formulate our DTIP following (1)-(10). The objective (1) is to minimize the total classification error over all the rows in the data:

$$\min \qquad \sum_{1\leq r\leq n} e_r \tag{1}$$

For every node in the tree, we use variable f_{ij} to indicate if feature i is used in the decision rule in node j. We use Equation (2) to make sure exactly one feature is used in each node:

$$s.t. \qquad \sum_{1\leq i\leq m} f_{ij} = 1, \qquad j = 1, ..., u \tag{2}$$

To encode all possible paths through the tree following the decision rules in each node, we use constraints (3) and (4):

$$\sum_{0\leq h<d(j)} M_L(r)\text{dlr}(h,j,r)+M_L(r)l_{d(j),r}+\sum_{1\leq i\leq m} v(r,i)f_{ij} \leq M_L(r)(d(j)+1)+c_j, \qquad j = 1, ..., u,\ r = 1, ..., n \tag{3}$$

$$\sum_{0\leq h<d(j)} M_R(r)\text{dlr}(h,j,r)-M_R(r)l_{d(j),r}-\sum_{1\leq i\leq m} v(r,i)f_{ij} \leq M_R(r)d(j)-c_j-1, \qquad j = 1, ..., u,\ r = 1, ..., n \tag{4}$$

Constraint (3) is a big-M formulation that makes sure the constraint only becomes active when a row r follows the path to node j and goes left at this node. The leftmost summation iterates over each layer and adds $M_L(r)$ only if row r follows the path to j in this layer. This means that when r ends up in node j, this term will add up to $M_L(r)d(j)$. The term $M_L(r)l_{d(j),r}$ adds another $M_L(r)$ if row r also goes left at node j. This means if row r passes through node j and goes left, the first two terms cancel with the term $M_L(r)(d(j)+1)$ on the right hand side, and we are left with the following inequality:

$$\sum_{1\leq i\leq m} v(r,i)f_{ij} \leq c_j$$

This constraint states that the sum of the feature values that are used in the decision rule in that node need to be below the threshold c_j. This constraint is thus only enforced when row r goes left at node j, as in all other cases, there will be at least one $M_L(r)$ not being cancelled out on the right side, in which case the equation always holds. Constraint (4) does the same, but now for rows that go right at node j, by restricting the corresponding feature value to be strictly greater than the threshold.

We make sure that every leaf corresponds to exactly one classification using constraint (5):

$$\sum_{t\in T} p_{lt} = 1, \qquad l = 1, ..., v \tag{5}$$

To compute the classification error for each row, we use constraint (6), that uses a similar

approach to constraints (3) and (4):

$$\sum_{0 \leq h < k} \text{dlr}(h, l, r) + \sum_{t \in T:\, t \neq t(r)} p_{lt} \leq e_r + k, \qquad l = 1, ..., v,\ r = 1, ..., n \tag{6}$$

On the left, we again have a summation to check if row r ends up in leaf l, but this time it will be equal to k if this is the case, and cancel with the k on the right-hand side. Our constraint then simplifies to the following inequality:

$$\sum_{t \in T:\, t \neq t(r)} p_{lt} \leq e_r$$

This left hand side in this inequality sums over each wrong classification and adds a value of 1 if this classification is in fact predicted by leaf l. This means it becomes 0 if leaf l makes the right prediction, and 1 if it makes the wrong prediction. The error is then enforced to be at least as great as this, but it follows from our objective that it is always optimal to set it equal to the left-hand side.

Next, we use constraints (7) and (8) to bound the thresholds to be between the lower bound and the upper bound of all feature values, in order to reduce the search space:

$$c_j \geq LF, \qquad j = 1, ..., u \tag{7}$$

$$c_j \leq UF, \qquad j = 1, ..., u \tag{8}$$

We add constraint (9) to make sure that two leaves from the same parent node give different classifications. If this is not be the case, the last split would not make much sense.

$$p_{lt} + p_{l't} \leq 1, \qquad t \in T \text{ and l and l' leaves of same parent} \tag{9}$$

Verwer and Zhang (2017) introduced this constraint as an equality, but this only works when there are exactly two different targets. In general, a certain target might not be predicted in either of the two leaves, so we change this into an inequality. Lastly, we bound our variables f_{ij}, l_{hr}, p_{lt} and e_r to be binary variables and we bound the thresholds c_j to integers. These constraints are made explicit in (10):

$$f_{ij} \in \mathbb{B}, \qquad i = 1, ..., m,\ j = 1, ..., u \tag{10}$$

$$c_j \in \mathbb{Z}, \qquad j = 1, ..., u$$

$$l_{hr} \in \mathbb{B}, \qquad h = 0, ..., k-1,\ r = 1, ..., n$$

$$p_{lt} \in \mathbb{B}, \qquad l = 1, ..., v,\ t \in T$$

$$e_r \in \mathbb{B}, \qquad r = 1, ..., n$$

3.2 Imbalanced DTIP (IDTIP)

In medical diagnosis, it is very important to be conservative when predicting if a patient is positive to a certain disease or not, as we do not want to wrongfully classify a healthy patient. We can discourage the decision tree to make such predictions by including an extra penalty on false positive predictions. We alter our regular DTIP formulation by adding a new constraint that counts the number of false positive classifications, which we store in the integer variable z:

$$z = \sum_{r:\, t(r)=0} e_r \tag{11}$$

The above formula is what this constraint would look like for the heart disease data, where 0 (no heart disease) is the negative class. For the breast cancer data, this negative class would be B (Benign). We include this term in our objective with a penalty size λ. The objective now looks as follows:

$$\min \qquad \sum_{1 \le r \le n} e_r + \lambda z \tag{12}$$

We will from now on refer to this imbalanced version of DTIP as IDTIP.

3.3 In-sample testing

We compute the decision trees with DTIP for depths 1 up to 5 using CPLEX (IBM) and compare the performance of DTIP with that of the classification method from scikit-learn (Pedregosa et al. (2011)), an optimized version of CART. Furthermore, we supply the CART solution as a warm start for our DTIP and we call this method DTIPs. Similarly, we refer to IDTIP with a warm start as IDTIPs. However, we do not consider IDTIP or IDTIPs for our in-sample analysis, as this will only reduce in-sample accuracy compared to regular DTIP and only becomes interesting when looking at out-of-sample performance.

Since CART is not guaranteed to produce complete trees, the solution it gives is not always feasible for our DTIP model and we need to make to make some manipulations before we can use it as a warm start. We add dummy nodes that hold constraints that are always met by setting the threshold to the UF. Using these dummy nodes and by adding dummy leaves, we can produce a complete decision tree. Finally, we make sure that every pair of leaves from the same parent have different classifications, by changing the right leaf to a different class. We can safely do this, because same classifications only occur when the parent node makes a redundant split, which we set to send all rows left. For each decision tree, we measure the performance based on the in-sample accuracy of the method, which is computed as the percentage of correctly classified instances.

3.4 Out-of-sample testing

In order to test the out-of-sample performances of the different methods on the breast cancer data and the heart disease data, we use 5-fold cross validation. This means that we train the model on a part of the sample, and use the resulting tree to classify the remaining instances. We apply this to different hold-out samples and compute the out-of-sample accuracy as the average

of the resulting accuracies. The reason we use 5 folds is because we observe large differences in accuracies between the different hold-out folds that can grow to over 10%. Therefore, we need multiple folds to get a reliable estimate of the out-of-sample accuracy. Taking into account that the running time of DTIP is usually half an hour, we decided to go with 5 folds, such that the total running time of this procedure is at maximum 2.5 hours for each instance. Because our breast cancer dataset consists of 569 observations, the first 4 folds will include 114 observations, and the last fold will include 113 observations. For the heart disease data, the first 2 folds consist of 60 observations, and the last 3 folds consist of 59 observations. We apply out-of-sample analysis on CART, DTIPs and IDTIPs for depths 1 up to 5. Here we only compare CART to the optimal decision tree methods with a warm start, as these always find solutions that are at least as good as regular DTIP and IDTIP solutions.

3.5 Parameter tuning

To optimize the out-of-sample performance of IDTIP, we apply parameter tuning on a depth of 1. This method runs relatively quickly, allowing us to investigate a wide range of penalty sizes. We use this analysis to get an idea of the influence of the penalty size and to choose the optimal penalty size λ for a depth of 1. However, when we increase the depth of the tree, the in-sample accuracy of the tree increases, thus increasing the relative importance of the penalty on false positives. Because of this, the optimal penalty size is different for each depth and therefore we apply further parameter tuning for the most promising depths to find the optimal parameter settings for IDTIP.

4 Results

4.1 In-sample analysis

The accuracies of the three methods CART, DTIP and DTIPs for decision tree depths varying between 1 and 5 can be found in Table 2. The time limit for each problem is set to 30 minutes. For DTIP and DTIPs, if no optimal solution was found after 30 minutes, we report the best feasible solution found by the CPLEX solver. This holds true for each of the runs we performed for DTIP, DTIPs, IDTIP and IDTIPs, both in-sample and out-of-sample.

The results for the "Iris", "Diabetes" and "Bank" data are very similar to those found by Verwer and Zhang (2017). We see that for the "Iris" dataset, DTIP and DTIPs always find the optimal solution, whereas CART only finds the optimal decision tree for depth 1, 2 and 5. For both of the other datasets, DTIP and DTIPs always find the optimal solution for a depth of 1, whereas CART does not find the optimal solution. However, for bigger instances DTIP does not structurally outperform CART when constructing trees of a depth higher than 1. Especially for the "Bank" data, we see that DTIP struggles constructing trees of depth 4 and 5, as it only achieves an accuracy of 11.52%. This shows that DTIP without a warm start is not an appropriate method when dealing with large datasets. Nonetheless, when we provide CPLEX with the CART solution as a warm start, as we do for DTIPs, we always obtain an accuracy that is at least as good, and in most cases better than that of the tree found by CART. There are some slight deviations between the accuracies we get and those obtained by Verwer and Zhang

(2017), but these only occur for instances that could not be solved to optimality, and can be attributed to the fact that we used a different computer to run the models.

Dataset	Method	Depth 1	Depth 2	Depth 3	Depth 4	Depth 5
Iris	CART	66.67%*	96%*	97.33%	99.33%	100%*
	DTIP	66.67%*	96%*	99.33%*	100%*	100%*
	DTIPs	66.67%*	96%*	99.33%*	100%*	100%*
Diabetes	CART	73.57%	77.21%	77.60%	79.17%	83.72%
	DTIP	75%*	77.73%	78.91%	74.35%	76.95%
	DTIPs	75%*	77.73%	79.43%	81.25%	84.38%
Bank	CART	88.50%	90.09%	90.47%	91.26%	92.08%
	DTIP	89.29%*	89.60%	84.67%	11.52%	11.52%
	DTIPs	89.29%*	90.09%	90.49%	91.26%	92.10%

Table 2: Classification accuracies of the different decision tree construction methods for depths 1-5. Values with a * indicate the optimal solutions.

The results of the in-sample analysis on the datasets regarding medical diagnosis can be found in Table 3. For both datasets, we see that all algorithms are able to find the optimal solution for a depth of 1. We again notice that DTIP outperforms CART on lower depths (depth 2), but performs worse than CART for higher depths (depth 3, 4 and 5 for "Breast Cancer" and depth 4 and 5 for "Heart Disease"). As before, when we consider DTIPs, we notice that it always performs at least as well as CART and usually results in better accuracies. What stands out in Table 3 are the acccuracies that are obtained for the "Breast Cancer" dataset. These grow up to 99.47% for a depth of 5, which is a clear sign of overfitting.

Dataset	Method	Depth 1	Depth 2	Depth 3	Depth 4	Depth 5
Breast Cancer	CART	92.27%*	94.20%	97.89%	98.24%	99.47%
	DTIP	92.27%*	95.96%	96.66%	93.32%	95.08%
	DTIPs	92.27%*	96.13%	97.89%	98.59%	99.47%
Heart Disease	CART	76.43%*	77.10%	85.52%	87.54%	91.25%
	DTIP	76.43%*	79.80%*	85.52%	86.53%	84.51%
	DTIPs	76.43%*	79.80%*	85.52%	89.23%	93.27%

Table 3: Classification accuracies of the different decision tree construction methods for depths 1-5 for the datasets on medical diagnosis. Values with a * indicate the optimal solutions.

4.2 Out-of-sample analysis

To properly analyse the performance of different models for the breast cancer and the heart disease data, we need to be aware of the risk of overfitting. For example, the breast cancer dataset consists of 569 observations and 30 features, which means that for deep trees we can achieve great in-sample accuracies, but these models typically do not work well on new observations. Therefore, we now move on to out-of-sample analysis, giving better insights in the predictive capabilities of the different models.

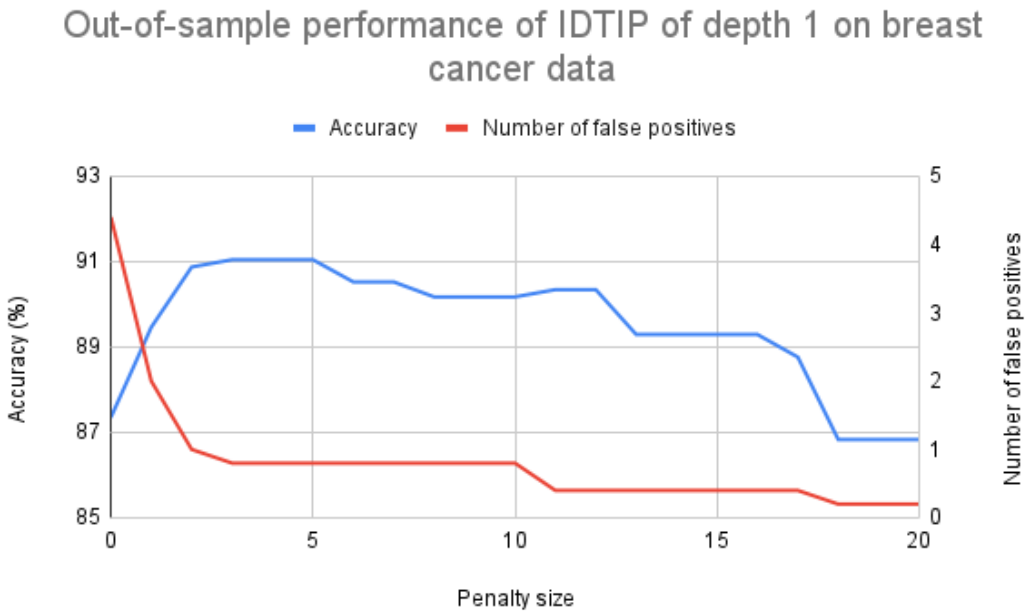

Figure 2: Out-of-sample accuracy in percentage (left axis) and average amount of out-of-sample false positives (right axis) for IDTIP of depth 1 with varying penalty sizes on breast cancer dataset

Parameter tuning for breast cancer data

We start by investigating the effect of parameter λ for IDTIP. Figure 2 shows the effect on the penalty size for IDTIP with depth 1 on the "Breast Cancer" dataset. From this figure, it is clear that a small penalty size helps to deal with the imbalance in the data, increasing the out-of-sample accuracy by up to 3.69% compared to regular DTIP. The reason for this is that inducing a small penalty on false positives results in a decision tree that produces more negative predictions, which corresponds with the structure of the data, since there are more negative than positive patients. When the penalty becomes too high, however, we notice that the tree becomes too conservative and produces too many false negative predictions, causing the accuracy to drop. The optimal out-of-sample accuracy is obtained for a penalty size of 3. For higher penalty sizes, we see no further increase in accuracy, and after a penalty size of 5, the accuracy starts to drop. In terms of the number of false positives, we see that the biggest reduction is made when moving from a penalty size of 0 to 1 and another large decrease is made when moving from 1 to 2. Again, we see that a penalty size of 3 is a turning point, after which the slope flattens.

The interpretation of this penalty is that every false positive prediction is penalized 4 times as much as a false negative prediction. To give an idea of how heavy of a penalty this is: the number of in-sample false positives over the whole sample is reduced from 11 false positives (and 33 false negatives) for no penalty, to 2 false positives (and 53 false negatives) for a penalty of size 3.

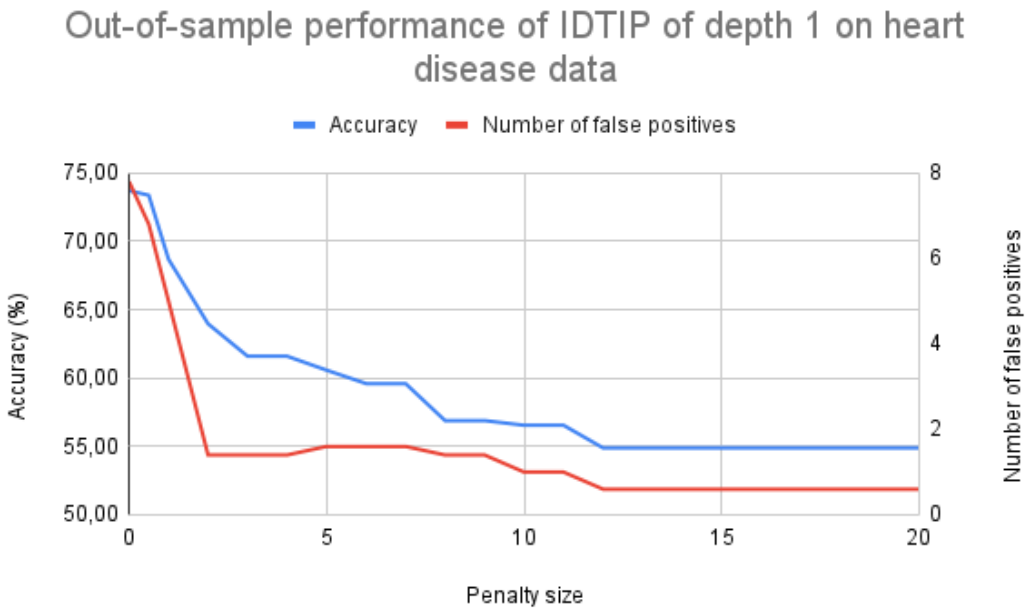

Figure 3: Out-of-sample accuracy in percentage (left axis) and average amount of out-of-sample false positives (right axis) for IDTIP of depth 1 with varying penalty sizes on heart disease dataset

Parameter tuning for heart disease data

Figure 3 shows the same analysis performed on the less imbalanced "Heart Disease" dataset. We see that for this dataset, the out-of-sample accuracy decreases for any penalty higher than 0. From this we can conclude that IDTIP for a depth of 1 does not improve the out-of-sample accuracy for this dataset, but it could still be used to confine the risk of making false positive predictions. For this purpose, a penalty size of 2 seems most fitting, as we see a step decline in number of false positives before this value, and barely any decline for higher penalties.

The interpretation of this penalty is that every false positive prediction is penalized 3 times as much as a false negative prediction. To put this into perspective: the number of in-sample false positives over the whole sample is reduced from 33 false positives (and 37 false negatives) for no penalty, to 4 false positives (and 109 false negatives) for a penalty of size 2.

Out-of-sample analysis on breast cancer data

The out-of-sample accuracies and the average amount of out-of-sample false positives for the "Breast Cancer" data computed by 5-fold cross validation for each of the methods are reported in Table 4. The first important observation is that for each of the methods, a decision tree of depth 2 (for DTIP) or 3 (for CART) gives the best accuracy, unlike in-sample, where decision trees of depth 5 always yielded the highest accuracy. Secondly we see that DTIPs can achieve an out-of-sample accuracy of 92.97% for the optimal depth, whereas the best out-of-sample accuracy for CART is only 91.38%. This shows that DTIPs does not only give better trees in terms of in-sample performance, but it also improves predictive performance compared to CART. To see if we can further improve these results by including a penalty on false positive

predictions, we apply IDTIPs for depth 2 and 3. We choose to investigate only these two depths, as we observe that at depth 1 the tree performs much worse than at all other depths. We also notice for both methods that the accuracy decreases for depths higher than 3, indicating that the model starts overfitting, making decision trees of depth 2 and 3 the most interesting models.

We compute the out-of-sample performance of IDTIPs for penalty sizes $\lambda = 1$, $\lambda = 2$ and $\lambda = 3$ and report the results in Table 5. We examine $\lambda = 3$ as a benchmark penalty size, as this was the optimal penalty size for a depth of 1, and we compare this to lower penalty sizes to account for the increase in in-sample accuracy of the model. Furthermore, we look into higher penalty sizes $\lambda = 5$ and $\lambda = 10$, to see if we can restrict the number of out-of-sample false positives. We now see that the best out-of-sample accuracy for DTIPs with a depth of 2 can be improved by including an additional penalty of 1 or 2 on every false positive prediction, both of which give the exact same results. In this way, we can achieve an out-of-sample accuracy of 93.15%, whilst only making 1.6 false positive out-of-sample predictions on average (0.88% of all predictions). We see that by further increasing the penalty for a depth of 2, we can slightly reduce the average number of false positives, from 1.6 to 1.4, against a small loss in out-of-sample accuracy. For a depth of 3, we find substantially higher numbers of out-of-sample false positive predictions for every penalty size. Furthermore, we see that an increase in penalty size does not necessarily result in less false positive predictions out-of-sample. For this dataset, it seems that the most effective way of controlling the number of false positive predictions is by choosing a low depth.

When we run IDTIPs with $\lambda = 1$ and $\lambda = 2$ for depth 2 over the whole sample, we again see that both methods produce the same decision trees. In both cases, we find the tree as shown in Figure 4. This classification tree has an in-sample accuracy of 95.43% and produces 2 false positive predictions over the entire sample (0.35% of all predictions).

Method	Measure	Depth 1	Depth 2	Depth 3	Depth 4	Depth 5
CART	Accuracy	87.35%	91.20%	**91.38%**	91.21%	91.21%
	#FP	4.4	6.2	4.8	5	5.2
DTIPs	Accuracy	87.35%	**92.97%**	92.79%	91.74%	90.86%
	#FP	4.4	3.2	4.8	4.2	4.6

Table 4: Out-of-sample accuracies and average number of out-of-sample false positives (#FP) computed by 5-fold cross validation for breast cancer data. Highest out-of-sample accuracy for each method is highlighted.

	Measure	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 5$	$\lambda = 10$
Depth 2	Accuracy	**93.15%**	**93.15%**	92.80%	92.80%	92.80%
	#FP	1.6	1.6	2.2	1.6	**1.4**
Depth 3	Accuracy	91.74%	91.56%	92.44%	92.01%	92.44%
	#FP	4.6	4.0	3.4	3.4	3.6

Table 5: Out-of-sample accuracies and average number of out-of-sample false positives (#FP) from IDTIPs with varying penalty sizes computed by 5-fold cross validation for breast cancer data. Highest out-of-sample accuracy and lowest average number of out-of-sample false positives are highlighted.

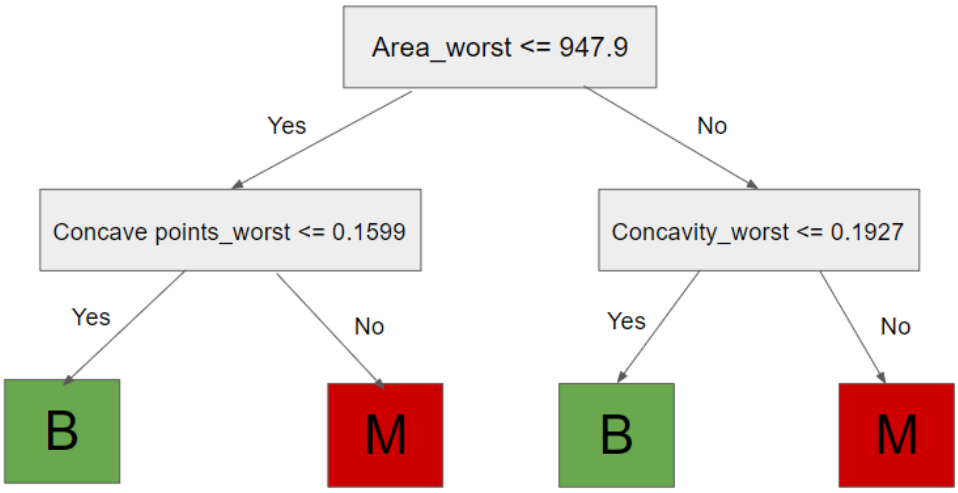

Figure 4: Best decision tree for breast cancer data, found with IDTIPs for depth 2 with penalty size $\lambda = 2$ over the whole sample (B = Benign, M = Malignant).

Out-of-sample analysis on heart disease data

The results of the same analysis for the "Heart Disease" data are reported in Table 6. From this table, we see that DTIPs again outperforms CART for the optimal depth, which is a depth of 3 for both of the methods. The out-of-sample accuracy is increased by almost 1%.

We again attempt to further improve these results by including a penalty on false positive predictions by applying IDTIPs for the most promising depth. We only investigate depth 3, due to the fact that this depth clearly leads to the best out-of-sample accuracies for both CART and DTIPs. We compare our benchmark penalty size $\lambda = 2$ to a lower penalty of $\lambda = 1$ to account for the improvement in in-sample accuracy of the model over a depth of 1, in order to potentially increase the out-of-sample accuracy. Furthermore, we look into higher penalty sizes $\lambda = 3$, $\lambda = 5$ and $\lambda = 10$, to see if we can restrict the number of out-of-sample false positives. These results can be found in Table 7. We find that IDTIPs performs over 5% worse than DTIPs in terms of out-of-sample accuracy for a penalty of 2, and hardly decreases the average number of false positive predictions. We can slightly improve this accuracy by moving to a penalty size of 1, but DTIP still performs better than IDTIPs in terms of out-of-sample accuracy. For the purpose of restricting the amount of false positives, we see that IDTIPs works for penalty $\lambda = 5$, as the

average amount of false positive predictions is reduced from 4 (6.7%) to 2.8 (4.7%). However, this is only a small improvement, at the cost of 7.4% in accuracy. Interestingly, we see that further increasing the penalty size to $\lambda = 10$ only leads to more out-of-sample false positives, as the model starts to overfit in terms of trying to reduce the number of false positives. In order to further decrease the average number of false positive predictions, we would need to turn to IDTIPs with a lower depth. As shown before in Figure 3, we can reduce the average amount of false positives to 0.6 (1.0%) by using a penalty size of 12 or higher for depth 1, but this does come at the cost of a massive loss of accuracy.

The best performing model for the heart disease data in terms of out-of-sample accuracy is DTIPs for depth 3. When we run this model over the whole dataset, we achieve an in-sample accuracy of 85.52% and 19 false positive predictions (6.4%). The resulting decision tree is visualized in Figure 5.

Method	Measure	Depth 1	Depth 2	Depth 3	Depth 4	Depth 5
CART	Accuracy	73.71%	73.05%	**80.13%**	76.76%	73.70%
	#FP	7.8	4.6	4.6	5.4	6.8
DTIPs	Accuracy	73.71%	73.03%	**81.12%**	72.73%	75.08%
	#FP	7.8	6	4	6	7

Table 6: Out-of-sample accuracies and average number of out-of-sample false positives (#FP) computed by 5-fold cross validation for heart disease data. Highest out-of-sample accuracy for each method is highlighted.

Measure	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 5$	$\lambda = 10$
Accuracy	**77.76%**	76.07%	74.72%	73.72%	67.32%
#FP	4.0	3.6	3.8	**2.8**	3.4

Table 7: Out-of-sample accuracies and average number of out-of-sample false positives (#FP) from IDTIPs of depth 3 with varying penalty sizes computed by 5-fold cross validation for heart disease data. Highest out-of-sample accuracy and lowest average number of out-of-sample false positives are highlighted.

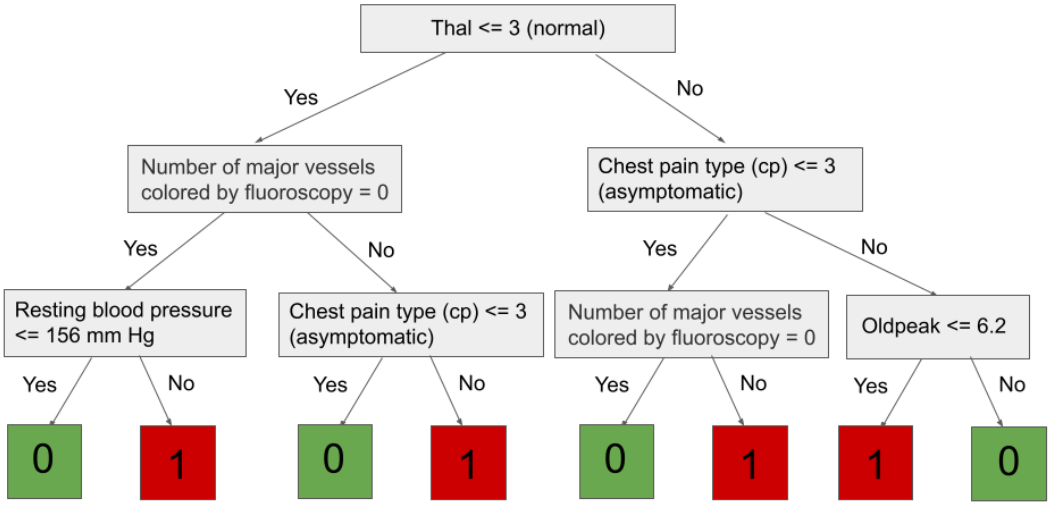

Figure 5: Best decision tree for heart disease data, found with DTIPs for depth 3 over the whole sample (0 = no heart disease present, 1 = heart disease present)

5 Conclusion

We investigated the potential of Decision Trees as Integer Programs (DTIP) for medical diagnosis purposes. We applied our methods to two datasets: one on breast cancer diagnosis, and the other on heart disease diagnosis, and investigated the in-sample and out-of-sample performances of our methods. We found that using DTIP with a CART solution as a warm start, we always find decision trees that have an in-sample accuracy greater or equal to that of CART. This especially works well on small datasets, such as the "Heart Disease" dataset, where we see improvements in in-sample accuracy of up to 2.7 percentage points. Furthermore, we found that DTIP with a warm start also improves the out-of-sample accuracy over CART for both of our datasets by up to 1.41 percentage points. This result shows that DTIP with a warm start is an improvement over greedy heuristics, not solely for the sake of constructing trees that fit the data better, but also for making predictions and diagnosing new patients. It demonstrates that the decision tree models produced by DTIP with a warm start resemble the true structure of the data more closely than those produced by greedy heuristics, such as CART.

We also investigated an imbalanced version of DTIP that penalizes false positive predictions more heavily, in order to restrict the number of healthy patients that are wrongfully diagnosed as unhealthy. After some parameter tuning, we were able to improve the out-of-sample accuracy whilst decreasing the average number of false-positive predictions for the "Breast Cancer" dataset, compared to regular DTIP. We found the most succesful method for this dataset to be imbalanced DTIP of depth 2 with a warm start and penalizing false positive predictions two or three times as heavily as false negative predictions. With this method we were able to achieve an out-of-sample accuracy of 93.15%. For the "Heart Disease" dataset, we were unable to improve the out-of-sample accuracy using imbalanced DTIP. The best performing decision tree for the "Heart Disease" data in terms of out-of-sample accuracy is found when applying

regular DTIP of depth 3 with a warm start, giving an out-of-sample accuracy of 81.12%. Restricting the number of false positive predictions is possible, but it comes at the cost of a loss in out-of-sample accuracy. To find a balance between this loss of accuracy and the number of false positive predictions, extensive parameter tuning is required, as increasing the penalty size does not necessarily result in fewer out-of-sample false positive predictions.

All in all, our results show that DTIP can be successfully applied in the context of medical diagnosis, as it provides easily interpretable models and outperforms heuristic methods to obtain such models. Furthermore, this flexible formulation allows for different objectives, which can be exploited to control the number of false positive predictions. Finding the optimal parameter settings is however quite expensive, due to the fact that each run of 5-fold cross validation takes 2.5 hours. For the purpose of medical diagnosis, these long running times should not be an issue, because these decision tree models only need to be trained once. If there happens to be a time shortage, one could resort to out-of-sample testing based on a single test set, lowering the running time limit, or any combination of these two.

As mentioned before, newer versions of optimal decision tree construction algorithms have been introduced over the last years, some of which are more efficient while offering the same flexibility as DTIP. These methods are expected to improve our results, as they can find better decision trees in the same amount of time. The applications of more recent optimal decision tree construction algorithms is therefore an interesting direction for further research. Another topic that could be of interest is the scalability of Decision Trees as Integer Programs. In this research, the focus was mainly on small datasets, but in general these tend to be much bigger. Investigating the performance of DTIP, or newer versions of optimal decision tree construction methods on large medical diagnosis datasets could therefore be valuable for future use of these algorithms.

References

S. Aghaei, A. Gómez, and P. Vayanos. Strong optimal classification trees, 2022.

K. Bennett and J. Blue. Optimal decision trees. 1996.

D. Bertsimas and J. Dunn. Optimal classification trees. *Machine Learning*, 106(7):1039–1082, 2017.

R. E. Bixby. A brief history of linear and mixed-integer programming computation. *Optimization Stories*, page 107–121, 2012.

L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. Taylor and Francis, 1984.

E. Demirović, A. Lukina, E. Hebrard, J. Chan, J. Bailey, C. Leckie, K. Ramamohanarao, and P. J. Stuckey. Murtree: Optimal classification trees via dynamic programming and search, 2020.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

E. Elyan and M. M. Gaber. A fine-grained random forests using class decomposition: an application to medical diagnosis. *Neural Computing and Applications*, 27:2279–2288, 2016.

H. Hu, M. Siala, E. Hebrard, and M.-J. Huguet. Learning optimal decision trees with maxsat and its integration in adaboost. In C. Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 1170–1176. International Joint Conferences on Artificial Intelligence Organization, 7 2020. Main track.

L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15–17, 1976.

IBM. IBM ILOG CPLEX Optimization Studio. URL `https://www.ibm.com/products/ilog-cplex-optimization-studio`.

A. Janosi, W. Steinbrunn, M. Pfisterer, and R. Detrano. Heart Disease. UCI Machine Learning Repository, 1988.

M. Kahn. Diabetes. UCI Machine Learning Repository, 1994.

J. Lin, C. Zhong, D. Hu, C. Rudin, and M. Seltzer. Generalized and scalable optimal sparse decision trees. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6150–6160. PMLR, 13–18 Jul 2020.

J. M. Luna, E. Eaton, L. H. Ungar, E. S. Diffenderfer, S. T. Jensen, E. D. Gennatas, M. Wirth, C. B. Simone, T. D. Solberg, and G. Valdes. Tree-structured boosting: Connections between gradient boosted stumps and full decision trees. *ArXiv*, 2017.

W. T. Mohammad, R. Teete, H. Al-Aaraj, Y. S. Rubbai, and M. M. Arabyat. Diagnosis of breast cancer pathology on the wisconsin dataset with the help of data mining classification and clustering techniques. *Applied Bionics and Biomechanics*, 2022:1–9, 2022.

S. Moro, P. Rita, and P. Cortez. Bank Marketing. UCI Machine Learning Repository, 2012.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

V. Podgorelec, P. Kokol, B. Stiglic, and I. Rozman. Decision trees: An overview and their use in medicine. *J. Med. Syst.*, 26(5):445–463, 2002.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

W. N. Street, W. H. Wolberg, and O. L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. *SPIE Proceedings*, 1993.

S. Verwer and Y. Zhang. Learning decision trees with flexible constraints and objectives using integer optimization. *Integration of AI and OR Techniques in Constraint Programming*, page 94–103, 2017.

S. Verwer and Y. Zhang. Learning optimal classification trees using a binary linear program formulation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1625–1632, 2019.

WHO. World Health Organization. URL `https://www.who.int/health-topics/cardiovascular-diseases`.

A Programming code

We used java as our main programming language for this project, from which we call the ILOG CPLEX optimization studio to run the optimization programs for DTIP. We also used python, in order to have acces to the scikit library to apply the optimized version of CART. The programming codes for the data transformation, CART, DTIP, DTIPs, IDTIP and IDTIPs can be found in the supplementary materials, alongside with all of our datafiles and a code manual. An explanation of how our code works and how we used it to run each of our experiments can be found in the code manual.