Tree-based Algorithms for Cardiovascular Disease Prediction

Mateusz Filipek[1]

[1]Silesian University Of Technology, Faculty of Applied Mathematics, Kaszubska 23, 44-100 Gliwice, Poland

Abstract

One of the issues data scientists run into the most frequently is the classification issue. We can separate the available data into discrete values with the use of classification. Numerous algorithms exist that enable us to solve this issue effectively. This article focuses on tree-based algorithms: Decision Tree Algorithm, and Random Forest Algorithm. The problem that is going to be approached with these algorithms is cardiovascular disease prediction, using the kaggle dataset containing records of patients data [1].

Keywords

Decision Trees, CART, Random Forest, Bagging algorithms

1. Introduction

Artificial intelligence methods [2, 3, 4] play an increasingly important role in various types of information systems. The numerous applications of artificial intelligence methods are based on several of its most important branches. One of the most important are methods based on fuzzy sets [5, 6]. In the papers [7, 8, 9] the authors proposed a system based on the second type fuzzy inference detecting anomalies on the roads. In the work [10], artificial intelligence methods based on fuzzy systems are responsible for the proper airing of rooms. The second very important branch of artificial intelligence algorithms are the [11, 12] heuristic algorithms, which are applicable wherever we strive to minimize or maximize functionals with different interpretations resulting from the specificity of the issue under consideration. At this point, it is worth paying attention to the work on reducing energy consumption [13, 14]. A very important group is the third branch of artificial intelligence methods based on neural networks [15, 16, 17, 18, 19]. They are used in many areas of life, including the detection of certain desirable features [20, 21], care for the elderly [22, 23, 24], diagnostics [25, 26, 27].

1.1. Cardiovascular Diseases

Each year approximately 175000 people in Poland die from cardiovascular diseases. The greatest cause of mortality worldwide, according to the WHO, are cardiovascular diseases, which claim approximately 17.9 million lives every year.

These numbers might go even higher in incoming years. The ongoing COVID pandemic, widespread lockdowns and the increase in people working from home can lead to increased numbers of people living sedentary lifestyles – and these can increase the likelihood of suffering from heart diseases.

Poor diet, inactivity, dangerous alcohol and tobacco use - these are just a few examples of lifestyle choices that can increase a person's chance of developing heart disease. Adult obesity is on the rise, and it's getting worse than ever. According to the CDC, US obesity prevalence increased from 30% to roughly 42%. Because cardiovascular diseases are responsible for roughly a third of global deaths, it is of the utmost importance to find a way to cure and help people who are suffering from heart diseases - but before such diseases can be treated properly, we need a way to detect them, hopefully long before they can do great harm.

Cardiovascular disease detection is a categorization problem. The results can often be split into two groups: healthy patients and patients with heart problems. Due to the fact that there are only two major result classes that can be simply defined using a binary system, such as 1 - a sick patient and 0 - a healthy patient, this particular classification task is known as binary classification.

1.2. Corrado Gini

Italian statistician Corrado Gini was born in Treviso, Italy, in 1884. Gini pursued his studies in law, mathematics, economics, and biology at the University of Bologna's Faculty of Law. Gini started out by looking into the connection between probability and population statistics. Later in his life, Gini's interest in demography studies led to the development of the theory of dispersion in "Variabilità e Mutabilità," which resulted in Gini's most well-known invention: the Gini coeficient, also known

ICYRIME 2022: International Conference of Yearly Reports on Informatics, Mathematics, and Engineering. Catania, August 26-29, 2022

matefil@polsl.pl (M. Filipek)

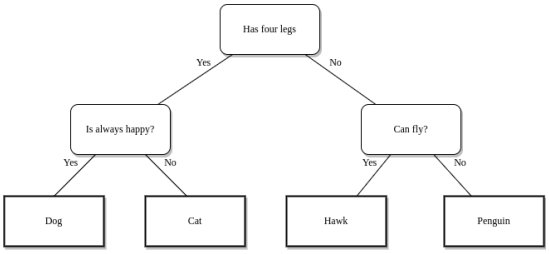

Figure 1: Example Decision Tree created using https://app.diagrams.net/ (https://app.diagrams.net/).

as Gini's Index, which is used to assess the degree of dispersion in a concentration.

1.3. Leo Breiman

American statistician Leo Breiman was born in New York, USA, in 1928. At the University of California, Leo Breiman pursued his education. Breiman is primarily recognized for his work on CART, Bootstrap Aggregation, which owes its name to him and is now known as Bagging. Leo Breiman is also the inventor of Random Forest method.

1.4. Binary Classification

The process of classifying the components of a set made up of only two classes is known as binary classification. The main applications of binary classification are in quality control and medical testing to check if a patient is ill or not, and to assess whether a produced thing fulfills the specification.

Some of the most common methods used for solving the binary classification problems are Decision Trees, Random Forests and Logistic Regression.

2. Proposed Classifiers

2.1. Decision Tree

Decision Trees are among the most commonly used models for classification and regression tasks. They can be described as a model whose purpose is to ask a dataset a list of if/else questions, and based on the responses the decision can be then made.

Decision Trees are often divided into two categories: classification and regression trees. Regression Trees produce numeric output, and classification trees produce categorical output. The latter is the main interest of this article, more specifically the CART implementation of decision trees.

The CART algorithm builds decision trees utilizing the Gini's Impurity Index to create best possible splits of data.

2.1.1. Gini's Impurity Index

The Gini's Ratio is a statistical dispersion metric that is frequently used to assess income disparity between countries. Gini's Impurity is a measure of the likelihood of choosing a certain feature that is incorrectly classified in decision trees. If all element in a dataset are of a single class, then Gini's Index takes the value of 0, meaning that that the dataset is pure. Similarly, if all elements of dataset are of different classes, then Gini's Index takes the value of 1, which indicates that the dataset is fully impure. If Gini's Index is of value 0.5, then the dataset is shows an equal distribution of elements over available classes.

Gini Index can be represented as:

$$\text{Gini Index} = 1 - \sum_{i=1}^{n} (P_i)^2,$$

Where P_i represents the probability of each element being classified for its distinct class.

In CART decision trees Gini's Index is used to calculate the best possible split at each level of the tree.

2.1.2. Algorithm

The Decision Tree Algorithm makes use of a binary tree data structure, where each node is either a decision node that is divided based on the best potential Gini Index, or a terminal node which does not further split, and decide about the predictions made by the decision tree. Decision Trees are often described using flowcharts.

The best possible split is calculated for each node individually, by checking all the possible values in each available feature. The pair of value + feature for which the best Gini's Index gain was achieved is used for splitting the dataset further into two parts.

Decision Tree is built and read recursively, thanks to the underlying data structure. Building the entire tree for a classifier involves using the training data that has been provided to determine the appropriate splits. It is important to adequately adjust the classifier parameters, such as maximum depth, or the minimum number of samples required for performing a split.

The maximum depth parameter specifies how deep the decision tree can get - it is the number of nodes from the root down to the furthest leaf node - the height of underlying tree structure.

Theoretically the maximum depth of the decision tree could be almost as high as the number of training samples, however it is not recommended to let the Decision Tree

Input TR: Training Samples, MaxDepth: Maximum Depth of the Decision Tree
Output Decision Tree built based on provided training samples **Building Decision Tree**

1: **if** stopping conditions are met **then**
2: $Return$ a leaf node with adequate class assigned to it
3: **else**
4: $gain =$ best possible Gini Gain
5: **if** $gain > 0$ **then**
6: recursively build left side of the current node
7: recursively build right side of the current node
8: $Return$ current node with both sides assigned to it

Predicting sample labels
Input xTest: testing samples
Output Predicted labels

1: **for do**
2: **if** currently checked node has a class value assigned to it **then**
3: $Return$ assigned class value
4: **else**
5: **if** currently tested node has key feature value greater than checked sample **then**
6: recursively check left child of node
7: recursively check right side of node

classifier grow to depth that high, because then it will overfit.

Data scientists use the term "overfitting" to indicate when the results of an analysis fit a set of data too closely. Such algorithm can perform very well on training data, but when exposed to an unknown sample, it will attempt to categorize it using highly specific criteria that may not be appropriate for classifying the unknown samples. The depth parameter should not be set too high because a model that has been trained too precisely on a given dataset will be fitted to that dataset exactly, which means it will learn not only how to make decisions based on the important features and their values but also how to take into account the existing "noise" - irrelevant information.

If there is more then a single sample present at a leaf node, then the outcome is predicted using the Majority Voting technique, where the class which has the highest number of representing samples is chosen.

2.2. Random Forest

Random Forest is an ensemble classification algorithm that performs classification using a predetermined number of decision trees. Random Forests rely on a lot of relatively unrelated - thanks to the randomly choosing of samples - trees, classifying the provided sample using each one and performing a majority vote to get the best possible result.

2.2.1. Ensemble Algorithms

The evaluation of the sample provided by ensemble algorithms typically requires more computing resources than it would for a single model, but the 'Wisdom of Crowds' obtained by using multiple models leads to increased accuracy.

2.2.2. Bagging

Bagging, also known as bootstrap aggregation, is an ensemble learning technique that is used to improve stability and accuracy, and to reduce variance within a dataset - decreasing the chance of overfitting models.

Using bagging, Random Forests produce a variety of trees by letting each one randomly select a sample from a given dataset. Creating a large number of decision trees helps in reducing overfitting.

2.2.3. Algorithm

Building a Random Forest
Input Training Samples, Number of trees
Output Random Forest built based on provided samples

1: **for** range(Number of trees) **do**
2: choose random subsample
3: create a Decision Tree using the chosen subsample

Classifying using Random Forest
Input Test Samples
Output Classified sample labels

1: **for** Every test sample **do**
2: **for** Every built tree **do**
3: Classify sample using currently checked Decision Tree
4: Perform majority voting based on results of classifying the chosen sample using all decision trees
5: $Return$ Classified samples

Each tree in Random Forest is built using randomly chosen data from dataset. When predicting the outcome of provided sample, the sample is provided to every available tree, and then the results from all the classifications are subjected to the Majority Voting technique, where the

	age	gender	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio	bmi
count	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000
mean	19468.865814	1.349571	128.817286	96.630414	1.366871	1.226457	0.088129	0.053771	0.803729	0.499700	27.556513
std	2467.251667	0.476838	154.011419	188.472530	0.680250	0.572270	0.283484	0.225568	0.397179	0.500003	6.091511
min	10798.000000	1.000000	-150.000000	-70.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	3.471784
25%	17664.000000	1.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000	23.875115
50%	19703.000000	1.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000	26.374068
75%	21327.000000	2.000000	140.000000	90.000000	2.000000	1.000000	0.000000	0.000000	1.000000	1.000000	30.222222
max	23713.000000	2.000000	16020.000000	11000.000000	3.000000	3.000000	1.000000	1.000000	1.000000	1.000000	298.666667

Figure 2: Dataset description (https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset).

most common outcome class is the output of the Random Forest algorithm.

Thanks to the use of multiple Decision Trees, Random Forest algorithms are usually highly effective,

3. The Cardiovascular Disease Dataset

The Cardiovascular Disease Dataset consists of 70000 records of patients data, consisting of 11 features each:

1. Age
2. Height
3. Weight
4. Gender
5. Systolic blood pressure
6. Diastolic blood pressure
7. Cholesterol
8. Glucose
9. Smoking
10. Alcohol intake
11. Physical activity

3.1. Data Cleaning

The process of detecting and fixing even removing, corrupted, duplicate, or incomplete data is known as data cleaning.

In the used dataset there is a number of invalid records, such as records of patients with systolic blood pressure that is negative or exceding 16000. Removal of such records allowed for reducing the total number of samples by 1413 records.

3.2. Dimensionality reduction

Dimensionality reduction is the process of minimizing the number of dimensions - features present in a dataset, while preserving the greatest amount of variety. Reducing the number of features accessible can increase performance, eliminate redundancy, and reduce overfitting.

Dimensionality reduction works by identifying and deleting elements that have little to no impact on the outcomes.

	age	gender	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio	bmi
age	1.00	-0.02	0.21	0.15	0.16	0.10	-0.05	-0.03	-0.01	0.24	0.10
gender	-0.02	1.00	0.06	0.07	-0.04	-0.02	0.34	0.17	0.01	0.01	-0.11
ap_hi	0.21	0.06	1.00	0.69	0.19	0.09	0.03	0.03	-0.00	0.43	0.26
ap_lo	0.15	0.07	0.69	1.00	0.16	0.07	0.02	0.04	0.00	0.34	0.23
cholesterol	0.16	-0.04	0.19	0.16	1.00	0.45	0.01	0.04	0.01	0.22	0.17
gluc	0.10	-0.02	0.09	0.07	0.45	1.00	-0.01	0.01	-0.01	0.09	0.12
smoke	-0.05	0.34	0.03	0.02	0.01	-0.01	1.00	0.34	0.03	-0.02	-0.03
alco	-0.03	0.17	0.03	0.04	0.04	0.01	0.34	1.00	0.02	-0.01	0.02
active	-0.01	0.01	-0.00	0.00	0.01	-0.01	0.03	0.02	1.00	-0.04	-0.01
cardio	0.24	0.01	0.43	0.34	0.22	0.09	-0.02	-0.01	-0.04	1.00	0.19
bmi	0.10	-0.11	0.26	0.23	0.17	0.12	-0.03	0.02	-0.01	0.19	1.00

Figure 3: Correlation Matrix

	age	ap_hi	ap_lo	cholesterol	cardio	bmi
count	68587.000000	68587.000000	68587.000000	68587.000000	68587.000000	68587.000000
mean	19462.927144	126.440710	81.312158	1.364063	0.494132	27.457441
std	2468.631074	16.355411	9.541735	0.678575	0.499969	5.258255
min	10798.000000	60.000000	40.000000	1.000000	0.000000	13.493001
25%	17656.000000	120.000000	80.000000	1.000000	0.000000	23.875115
50%	19700.000000	120.000000	80.000000	1.000000	0.000000	26.346494
75%	21323.000000	140.000000	90.000000	1.000000	1.000000	30.116213
max	23713.000000	197.000000	190.000000	3.000000	1.000000	74.380165

Figure 4: Dataset after dimensionality reduction and data cleaning

	age	ap_hi	ap_lo	cholesterol	cardio	bmi
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
mean	0.541637	0.420536	0.412810	0.192500	0.511000	0.344680
std	0.270335	0.145027	0.093931	0.348303	0.500129	0.153818
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.335999	0.363636	0.400000	0.000000	0.000000	0.235683
50%	0.569655	0.363636	0.400000	0.000000	1.000000	0.311390
75%	0.751204	0.545455	0.500000	0.500000	1.000000	0.429567
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Figure 5: Normalized data

3.2.1. Correlation Matrix

The correlation coefficients for each variable in the dataset are shown in a table called a correlation matrix. Such a table displays the association between each pair of attributes in each cell.

Correlation Matrix can be used to identify least important features in dataset, allowing for easy dimensionality reduction.

3.3. Normalization

Normalization is the process of scaling data to fit given range.

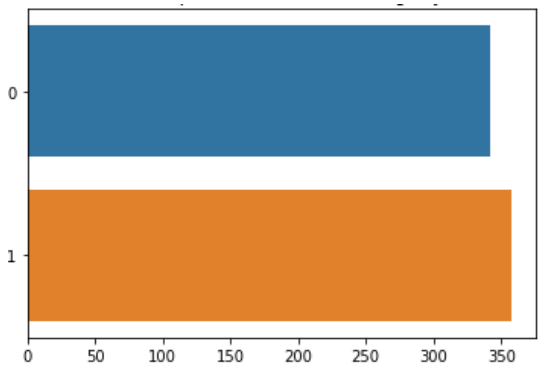

Figure 6: Ratio of classes in training portion of data set

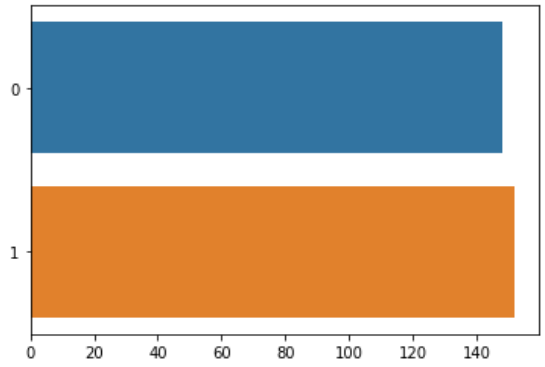

Figure 7: Ratio of classes in testing portion of data set

4. Classification

4.1. Splitting data

Dataset needs to be split into the training data, and testing data. Most of the data should be used for training purposes.

After a model has been trained using the training set, it's accuracy can then be validated using data from testing set. Because class values in testing set are already known, the accuracy of classifiers can be correctly calculated.

5. Classifier Evaluation

5.1. Correctness measures

5.1.1. P

P is the number of real positive conditions

5.1.2. N

N is the number of real negative conditions

5.1.3. TP

TP is the number of correctly predicted presence of a condition.

5.1.4. TN

TN is the number of correctly predicted absence of a condition

5.1.5. FP

FP is the number of wrongly predicted presence of a condition

5.1.6. FN

FN is the number of wrongly predicted absence of a condition

These four correctness measure metrics are the parameters of confusion matrix, they are used to evaluate specificity, sensitivity and accuracy of classifiers.

5.1.7. Accuracy

The simplest evaluation metric is accuracy. It measures how accurately projected classes compare to the whole testing dataset size. The number of labels that were successfully assigned is known as accuracy.

$$\text{ACC} = \frac{TP + TN}{P + N}$$

5.1.8. Precision

Precision is defined as the ratio of true positives to the sum of true positives and false positives. Precision describes how effectively the model predicts the positive cases out of all the cases it predicts as being true.

$$\text{PPV} = \frac{TP}{TP + FP}$$

5.1.9. Recall

The proportion of genuine positives to the total of true positives and false negatives is known as recall. Recall demonstrates how well the model separates out the positive cases from all the positive cases in the dataset.

$$\text{TPR} = \frac{TP}{TP + FN}$$

5.1.10. F1

The harmonic mean of recall and precision is the F_1 score.

$$F_1 = 2 * \frac{PPV * TPR}{PPV + TPR}$$

5.1.11. Confusion Matrix

Confusion Matrix is a performance measurement technique, as the name suggests - it is a matrix, representing four different combinations of predicted and actual values. Its name comes from the fact that using this matrix makes it easier to determine whether the model is incorrectly classifying classes.

$$\begin{pmatrix} TP & FP \\ FN & TN \end{pmatrix}$$

6. Testing the Decision Tree Classifier

The main deciding parameter in Decision Tree Classifier that needs to be adjusted is the maximum depth. Correctness measured for Decision Tree with maximum depth of 5 were:

ACC = 0.66
PPV = 0.76
TPR = 0.65

Correctness measured for Decision Tree with maximum depth of 10 were:

ACC = 0.61
PPV = 0.62
TPR = 0.63

Correctness measured for Decision Tree with maximum depth of 50 were:

ACC = 0.61
PPV = 0.62
TPR = 0.63

6.0.1. Conclusions

Single Decision Trees quickly began to overfit, increasing the maximum depth not only decreased the correctness of it's predictions, but also increased the total time needed for building the tree. The problem of overfitting can be fixed by utilizing the bagging technique.

7. Testing the Random Forest Classifier

The main parameters in Random Forest Classifier that needs to be adjusted are the total number of trees, as well as the maximum depth of every single tree.

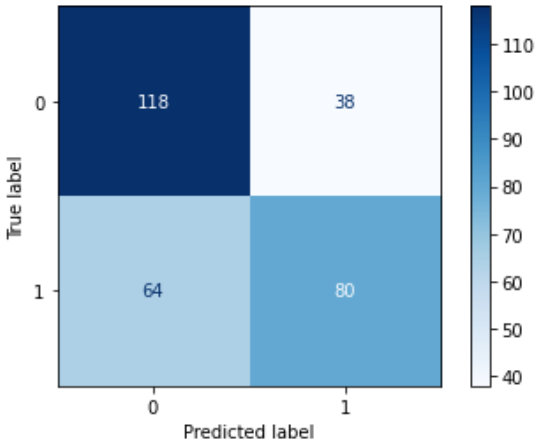

Figure 8: CM of Decision Tree Classifier, maximum depth = 5

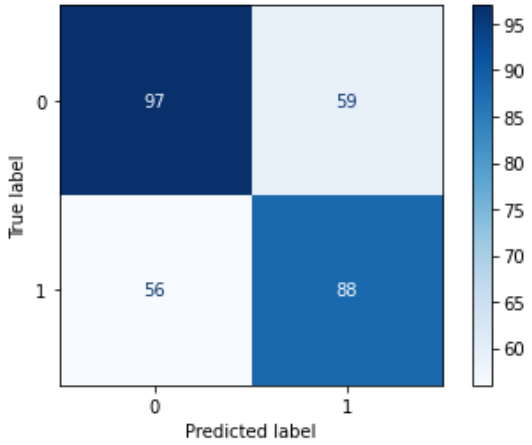

Figure 9: CM of Decision Tree Classifier, maximum depth = 10

7.0.1. Testing various numbers of trees

Correctness measured for Random Forest Classifier with total number of trees equal to 10, and the maximum depth of trees equal to 10 were:

ACC = 0.65
PPV = 0.74
TPR = 0.62

Correctness measured for Random Forest Classifier with total number of trees equal to 25, and the maximum depth of trees equal to 10 were:

ACC = 0.67
PPV = 0.79

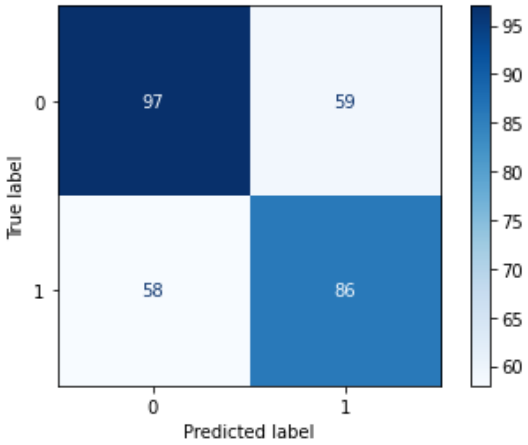

Figure 10: CM of Decision Tree Classifier, maximum depth = 50

```
For 2 depth, the time needed for predictions was 0:22:33.525572
For 4 depth, the time needed for predictions was 0:35:08.601088
For 8 depth, the time needed for predictions was 0:53:51.319691
For 16 depth, the time needed for predictions was 1:05:05.059329
For 32 depth, the time needed for predictions was 1:07:07.889358
For 64 depth, the time needed for predictions was 1:05:35.035823
For 128 depth, the time needed for predictions was 1:09:19.323658
```

Figure 11: Time needed for predictions in Decision Tree Classifier

TPR = 0.63

Correctness measured for Random Forest Classifier with total number of trees equal to 50, and the maximum depth of trees equal to 10 were:

ACC = 0.66
PPV = 0.74
TPR = 0.63

Correctness measured for Random Forest Classifier with total number of trees equal to 100, and the maximum depth of trees equal to 10 were:

ACC = 0.68
PPV = 0.79
TPR = 0.64

7.0.2. Conclusions

Increasing the total number of trees increased the correctness of Random Forest classifier. Even with low total amount of trees Random Forest classifier has better correctness than a single Decision Tree, bagging helps with overfitting, the choosing of random samples helps the classifier to learn the training dataset better.

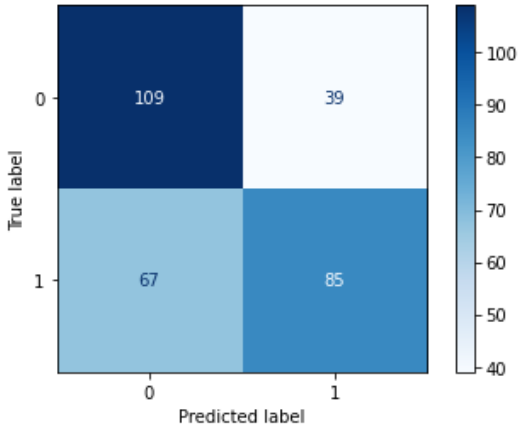

Figure 12: CM of Random Forest Classifier, maximum depth = 10, total number of trees = 10

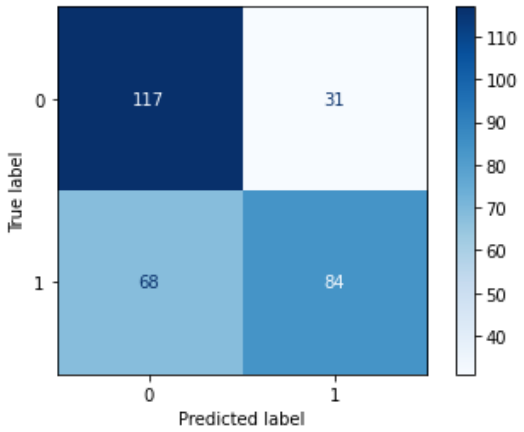

Figure 13: CM of Random Forest Classifier, maximum depth = 10, total number of trees = 25

7.0.3. Testing various maximum depth

Correctness measured for Random Forest Classifier with total number of trees equal to 25, and the maximum depth of trees equal to 2 were:

ACC = 0.66
PPV = 0.74
TPR = 0.64

Correctness measured for Random Forest Classifier with total number of trees equal to 25, and the maximum depth of trees equal to 4 were:

ACC = 0.67
PPV = 0.79
TPR = 0.63

Correctness measured for Random Forest Classifier

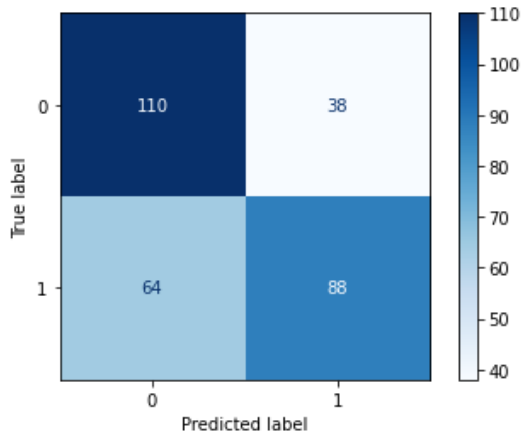

Figure 14: CM of Random Forest Classifier, maximum depth = 10, total number of trees = 50

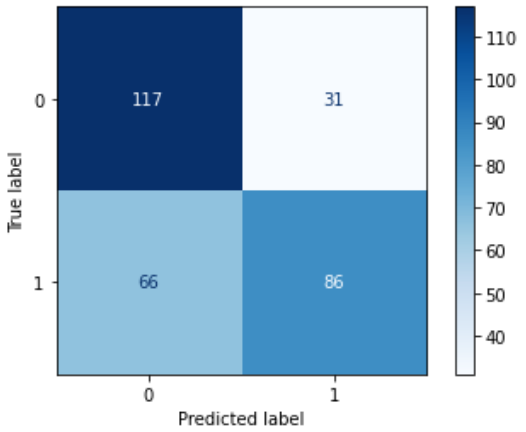

Figure 15: CM of Random Forest Classifier, maximum depth = 10, total number of trees = 100

```
For 10 trees, the time needed for predictions was 0:19:36.247299
For 25 trees, the time needed for predictions was 0:48:36.629862
For 50 trees, the time needed for predictions was 1:36:39.952676
For 100 trees, the time needed for predictions was 3:12:18.024596
```

Figure 16: Time needed for predictions in Random Forest Classifier

with total number of trees equal to 25, and the maximum depth of trees equal to 16 were:

ACC = 0.6
PPV = 0.61
TPR = 0.59

Correctness measured for Random Forest Classifier with total number of trees equal to 25, and the maximum

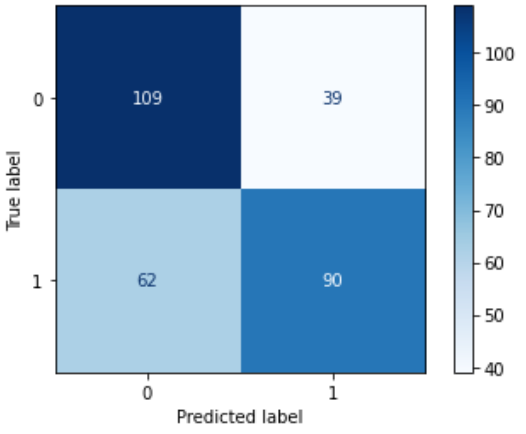

Figure 17: CM of Random Forest Classifier, maximum depth = 2, total number of trees = 25

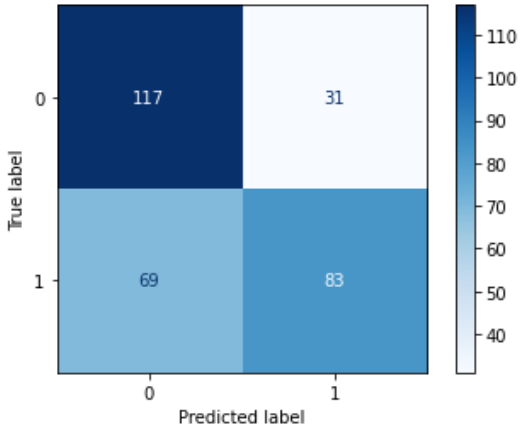

Figure 18: CM of Random Forest Classifier, maximum depth = 4, total number of trees = 25

depth of trees equal to 128 were:

ACC = 0.62
PPV = 0.61
TPR = 0.61

7.0.4. Conclusions

Changes in the maximum depth of individual trees did not affect the Random Forest classifier as much as the total amount of trees.

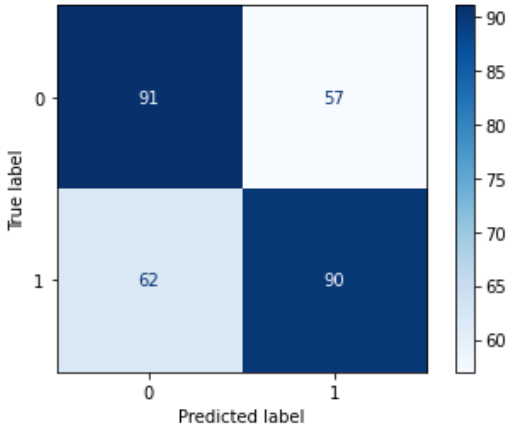

Figure 19: CM of Random Forest Classifier, maximum depth = 16, total number of trees = 25

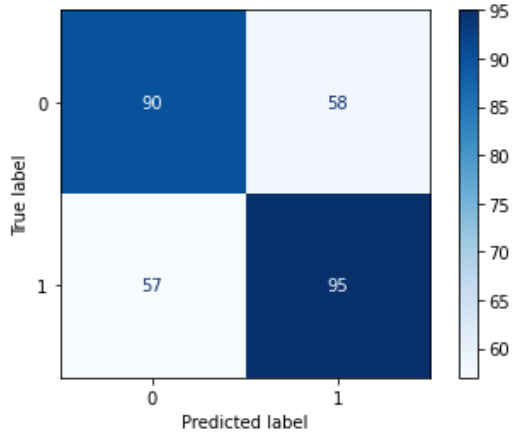

Figure 20: CM of Random Forest Classifier, maximum depth = 128, total number of trees = 25

8. Conclusions

The nature of disease prediction problem makes it suitable to use tree-based algorithms for patient classification. The proposed algorithms show good correctness. The presented tests have shown that the proper selection of classifiers has a great effect on the classification results. Decision Trees alone can predict reasonably well, however utilizing bagging algorithms such as Random Forest can increase the correctness of acquired results, sacrificing a little execution speed.

References

[1] S. Ulianova, Cardiovascular disease dataset, 2019. URL: https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset.

[2] Q.-b. Zhang, P. Wang, Z.-h. Chen, An improved particle filter for mobile robot localization based on particle swarm optimization, Expert Systems with Applications 135 (2019) 181–193.

[3] M. A. Sanchez, O. Castillo, J. R. Castro, Generalized type-2 fuzzy systems for controlling a mobile robot and a performance comparison with interval type-2 and type-1 fuzzy systems, Expert Systems with Applications 42 (2015) 5904–5914.

[4] V. Ponzi, S. Russo, V. Bianco, C. Napoli, A. Wajda, Psychoeducative social robots for an healthier lifestyle using artificial intelligence: a case-study, in: CEUR Workshop Proceedings, volume 3118, 2021, pp. 26–33.

[5] Y. Li, W. Dong, Q. Yang, S. Jiang, X. Ni, J. Liu, Automatic impedance matching method with adaptive network based fuzzy inference system for wpt, IEEE Transactions on Industrial Informatics 16 (2019) 1076–1085.

[6] Y. Sun, H. Qiang, J. Xu, G. Lin, Internet of things-based online condition monitor and improved adaptive fuzzy control for a medium-low-speed maglev train system, IEEE Transactions on Industrial Informatics 16 (2020) 2629–2639. doi:10.1109/TII.2019.2938145.

[7] M. Woźniak, A. Zielonka, A. Sikora, Driving support by type-2 fuzzy logic control model, Expert Systems with Applications 207 (2022) 117798.

[8] N. Brandizzi, S. Russo, R. Brociek, A. Wajda, First studies to apply the theory of mind theory to green and smart mobility by using gaussian area clustering, in: CEUR Workshop Proceedings, volume 3118, 2021, pp. 71–76.

[9] N. Brandizzi, S. Russo, G. Galati, C. Napoli, Addressing vehicle sharing through behavioral analysis: A solution to user clustering using recency-frequency-monetary and vehicle relocation based on neighborhood splits, Information (Switzerland) 13 (2022). doi:10.3390/info13110511.

[10] M. Woźniak, A. Zielonka, A. Sikora, M. J. Piran, A. Alamri, 6g-enabled iot home environment control using fuzzy rules, IEEE Internet of Things Journal 8 (2020) 5442–5452.

[11] T. Qiu, B. Li, X. Zhou, H. Song, I. Lee, J. Lloret, A novel shortcut addition algorithm with particle swarm for multisink internet of things, IEEE Transactions on Industrial Informatics 16 (2019) 3566–3577.

[12] D. Yu, C. P. Chen, Smooth transition in communication for swarm control with formation change, IEEE

Transactions on Industrial Informatics 16 (2020) 6962–6971.

[13] M. Woźniak, A. Sikora, A. Zielonka, K. Kaur, M. S. Hossain, M. Shorfuzzaman, Heuristic optimization of multipulse rectifier for reduced energy consumption, IEEE Transactions on Industrial Informatics 18 (2021) 5515–5526.

[14] V. Ponzi, S. Russo, A. Wajda, R. Brociek, C. Napoli, Analysis pre and post covid-19 pandemic rorschach test data of using em algorithms and gmm models, in: CEUR Workshop Proceedings, volume 3360, 2022, pp. 55–63.

[15] V. S. Dhaka, S. V. Meena, G. Rani, D. Sinwar, M. F. Ijaz, M. Woźniak, A survey of deep convolutional neural networks applied for prediction of plant leaf diseases, Sensors 21 (2021) 4749.

[16] S. Pepe, S. Tedeschi, N. Brandizzi, S. Russo, L. Iocchi, C. Napoli, Human attention assessment using a machine learning approach with gan-based data augmentation technique trained using a custom dataset, OBM Neurobiology 6 (2022). doi:10.21926/obm.neurobiol.2204139.

[17] F. Bonanno, G. Capizzi, G. Lo Sciuto, A neuro wavelet-based approach for short-term load forecasting in integrated generation systems, in: 2013 International Conference on Clean Electrical Power (ICCEP), IEEE, 2013, pp. 772–776.

[18] G. Capizzi, C. Napoli, L. Paternò, An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys, in: Artificial Intelligence and Soft Computing: 11th International Conference, ICAISC 2012, Zakopane, Poland, April 29-May 3, 2012, Proceedings, Part I 11, Springer, 2012, pp. 21–29.

[19] G. Lo Sciuto, G. Susi, G. Cammarata, G. Capizzi, A spiking neural network-based model for anaerobic digestion process, in: 2016 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), IEEE, 2016, pp. 996–1003.

[20] O. Dehzangi, M. Taherisadr, R. ChangalVala, Imu-based gait recognition using convolutional neural networks and multi-sensor fusion, Sensors 17 (2017) 2735.

[21] R. Aureli, N. Brandizzi, G. Magistris, R. Brociek, A customized approach to anomalies detection by using autoencoders, in: CEUR Workshop Proceedings, volume 3092, 2021, pp. 53–59.

[22] S. Russo, S. Illari, R. Avanzato, C. Napoli, Reducing the psychological burden of isolated oncological patients by means of decision trees, in: CEUR Workshop Proceedings, volume 2768, 2020, pp. 46–53.

[23] S. Russo, C. Napoli, A comprehensive solution for psychological treatment and therapeutic path planning based on knowledge base and expertise sharing, in: CEUR Workshop Proceedings, volume 2472, 2019, pp. 41–47.

[24] M. Woźniak, M. Wieczorek, J. Siłka, D. Połap, Body pose prediction based on motion sensor data and recurrent neural network, IEEE Transactions on Industrial Informatics 17 (2020) 2101–2111.

[25] G. Lo Sciuto, S. Russo, C. Napoli, A cloud-based flexible solution for psychometric tests validation, administration and evaluation, in: CEUR Workshop Proceedings, volume 2468, 2019, pp. 16–21.

[26] H. G. Hong, M. B. Lee, K. R. Park, Convolutional neural network-based finger-vein recognition using nir image sensors, Sensors 17 (2017) 1297.

[27] S. Illari, S. Russo, R. Avanzato, C. Napoli, A cloud-oriented architecture for the remote assessment and follow-up of hospitalized patients, in: CEUR Workshop Proceedings, volume 2694, 2020, pp. 29–35.