Classificatikon of Obesity Types using Random Forest and Decision Tree algorithms[*]

Martyna Kramarz[1,*,†], Dominik Dzida[1,†]

[1]Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, POLAND

Abstract
In this paper, we focused on usage of two algorithms: Decision Tree and Random Forest and analyzing dataset that contains blood work of adults and adolescence. Dataset consists of 17 columns including last one that classifies in one of six categories of illness : Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Insufficient Weight, Obesity Type II, Obesity Type III. We researched which algorithms were used in similar datasets, and picked those that were most compatible. After data processing and modification to achieve the maximum accuracy. We managed finished with Random Forest with accuracy at 97 percent compared to Decision Tree 95 percent.LATEX.

Keywords
Machine Learning, Data Analysis, Obesity Prediction, Feature Engineering

1. Introduction

Obesity is an enormous problem and the number of people affected by it is growing every day[1]. In 2022, 2.5 billion adults aged 18 years and older were overweight, including over 890 million adults who were living with obesity[2]. Obesity can result in serious health issues that are potentially life threatening, including hypertension, type II diabetes mellitus, increased risk for coronary disease, increased unexplained heart failure, hyperlipidemia, infertility, higher prevalence of colon, prostate, endometrial, and breast cancer [3, 4]. Using algorithms to fasten the process of classification of medical problems, can have great results which has been proven to work in many areas: brain tumor [4, 6], neurodegenerative disorder [5], malaria detection [7] and many more.

The subject of obesity was also covered in literature and it was established that ML algorithms can work greatly in self-management of obesity and integration into electronic tools such as mobile devices and EHRs, for personal or population-based clinical decision-making, can be explored by researchers towards the development of smart and impactful digital health interventions [8,9]. The decision of choosing this specific algorithms was based on knowledge that both algorithms have been tested on similar dataset (one also related to determining degree of childhood obesity) [10] and analyzing the inner-working of the algorithms.

[*]IVUS2024: Information Society and University Studies 2024, May 17, Kaunas, Lithuania
[1,*]Corresponding author
[†] These author contributed equally.

mk307890@student.polsl.pl (M. Kramarz); dd307860@student.polsl.pl (D. Dzida)
0009-0007-8281-679X (M. Kramarz); 0009-0008-6655-6155 (D. Dzida)

Decision Trees is a supervised classification approach that consists of similar elements to ordinary tree structure: a root and nodes. A Decision Tree starts from the root, moves downward and generally is drawn from left to right. The node from where the tree starts is called a root node. The node where the chain ends is known as the “leaf” node. A node represents a certain characteristic while the branches represent a range of values. These ranges of values act as a partition points for the set of values of the given characteristic [11].

Random Forest is an ensemble learning method used for classification and regression tasks that constructs multiple decision trees during training. It combines the predictions from these trees to improve accuracy and control over-fitting. Each tree in the forest is built from a random subset of the training data, and features are randomly selected for splitting at each node. This randomness leads to a diverse set of models, whose collective output is more robust and generalizes better to unseen data [12, 13].

2. Dataset

In our work we were using dataset form site kaggle called Obesity or CVD risk. The dataset comprises estimates of obesity levels among individuals from Mexico, Peru, and Colombia, aged between 14 and 61, with varied eating habits and physical conditions. The data was gathered through a web-based survey where anonymous participants responded to each question. The collected information was then processed, resulting in 17 attributes and 2111 records.

2.1. Dataset column description

Our data set consists of 17 columns which are:

- Gender - text value "Male" or "Female"
- Age - numerical value from 14 to 61 (in years)
- Height - comma numeric value from 1.45 to 1.98 (in meters)
- Weight - comma numeric value from 39 to 173 (in kilograms)
- family_history_with_overweight - text value "yes" or "no" - informs if family member suffered or suffers from overweight
- FAVC - text value "yes" or "no" - frequent consumption of high caloric food
- FCVC - numerical value from 1 to 3 - frequency of consumption of vegetables
- NCP - numerical value from 1 to 4 - number of main meals
- CAEC - text value "Always", "Frequently", "Sometimes" or "no" - consumption of food between meals
- SMOKE - text value "yes" or "no" - smoker or not
- CH2O - numerical value from 1 to 3 (in liters) - consumption of water daily
- SCC - text value "yes" or "no" - calories consumption monitoring
- FAF - numerical value from 0 to 3 - physical activity frequency
- TUE - numerical value from 0 to 2 - time using technology devices
- CALC - text value "no", "Sometimes", "Frequently" or "Always" - consumption of alcohol

- MTRANS - text value "Public_Transportation", "Walking", "Automobile", "Motorbike" or "Bike" - transportation used
- NObeyesdad "Normal_Weight", "Overweight_Level_I", "Overweight_Level_II", "Obesity_Type_I", "Insufficient_Weight", "Obesity_Type_II" or "Obesity_Type_III" - text value - obesity level deducted

2.2. Data preparation

To make working with the data easier, we decided to present all information in numerical form.We used the conversion for the following columns: "NObeyesdad", "MTRANS", "CALC", "SCC", "SMOKE", "CAEC", "FAVC", "family_history_with_overweight" and "Gender".We made the conversion in accordance with the tables below.

Table 1
MTRANS

Text value	Numeric value
Public_Transportation	4
Walking	3
Automobile	2
Motorbike	1
Bike	0

Table 2
NObeyesdad

Text value	Numeric value
Normal_Weight	6
Overweight_Level_I	5
Overweight_Level_II	4
Obesity_Type_I	3
Insufficient_Weight	2
Obesity_Type_II	1
Obesity_Type_III	0

Table 3
CALC

Text value	Numeric value
Always	3
Frequently	2
Sometimes	1
no	0

Table 4
CAEC

Text value	Numeric value
Always	3
Frequently	2
Sometimes	1
no	0

Table 5
SCC

Text value	Numeric value
no	1
yes	0

Table 6
SMOKE

Text value	Numeric value
no	1
yes	0

Table 7
FAVC

Text value	Numeric value
yes	1
no	0

Table 8
family_history_with_overweight

Text value	Numeric value
yes	1
no	0

Table 9
Gender

Text value	Numeric value
Female	1
Male	0

2.3. Data analysis

After changing all values to numerical values, we checked whether there were no columns in our set containing empty values. After confirming the correctness of all our data, we decided to check the distribution of this data into individual categories in each column of our dataset.Below are charts describing this distribution:

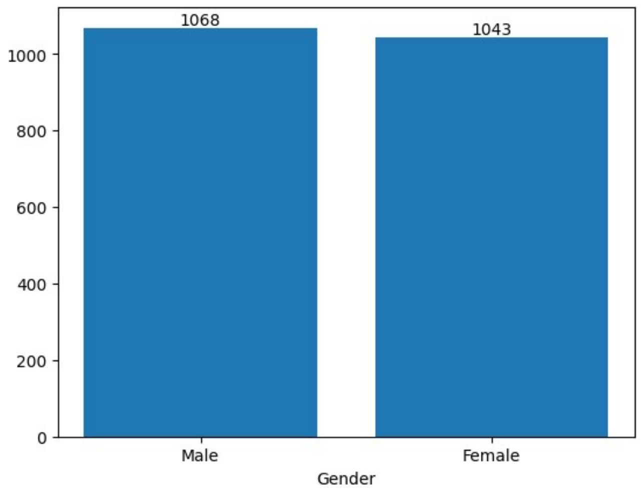


Figure 1: Gender distribution.

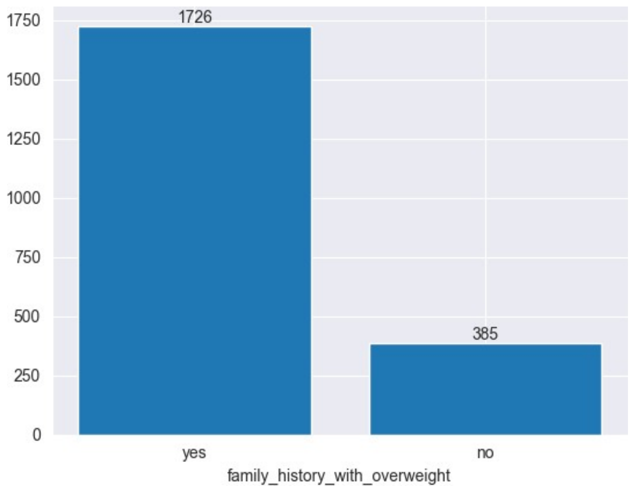


Figure 2: Family history with overweight distribution.

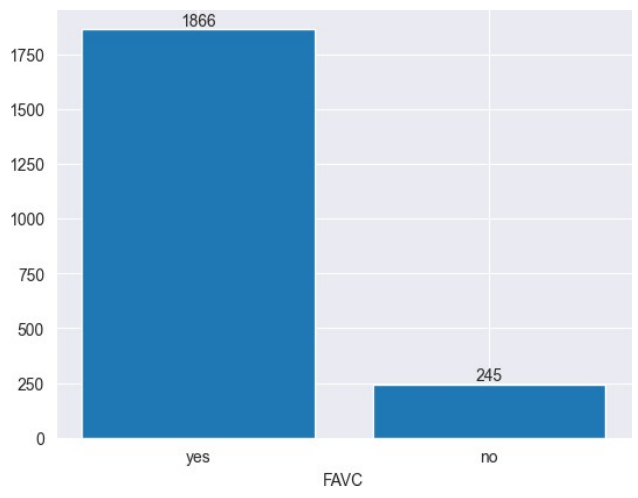


Figure 3: FAVC distribution.

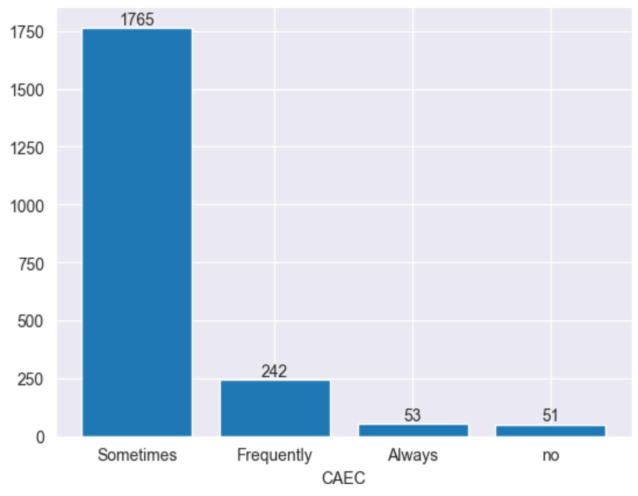


Figure 4: CAEC distribution.

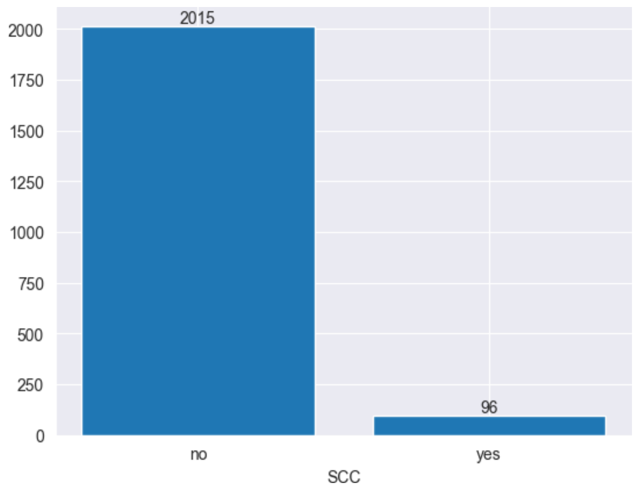


Figure 5: SCC distribution.

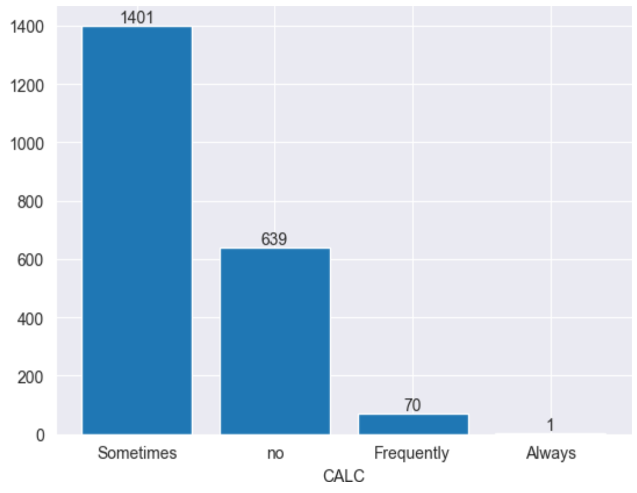


Figure 6: CALC distribution.

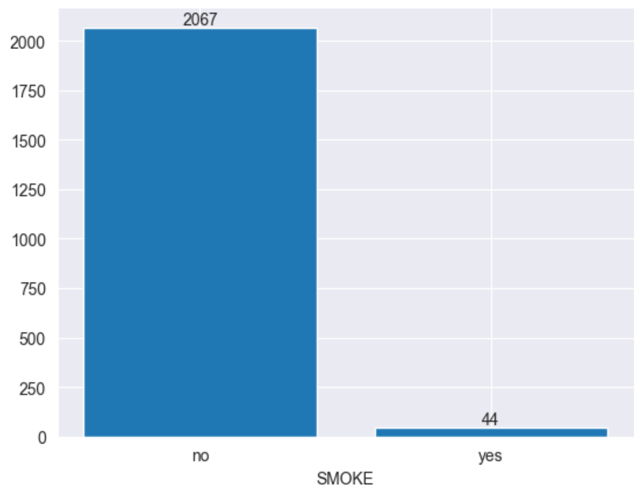


Figure 7: Smoking distribution.

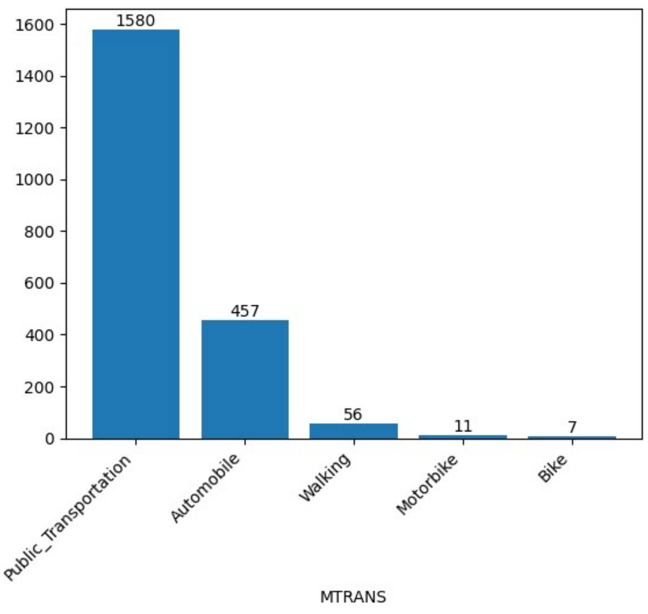


Figure 8: MTRANS distribution.

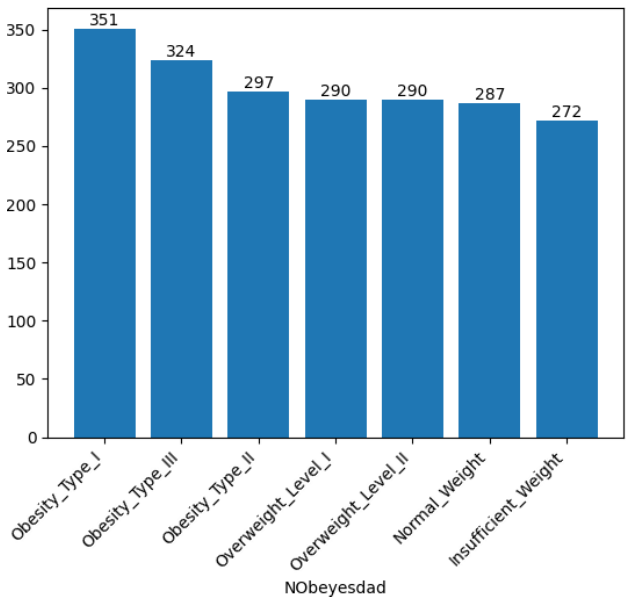


Figure 9: NObeyesdad distribution.

3. Algorithms

3.1. Selection of algorithms

In order to select an appropriate classifier for our set, we used the ready-made sklearn library containing a significant number of classifiers. After comparing the accuracy results of many of them, we chose Decision Tree and Random Forest due to their high accuracy.

3.2. Decision Tree

3.2.1. Quick description of the Decision Tree algorithm

The decision tree algorithm is a machine learning method used for classification and regression. It involves the iterative division of a data set into subsets based on attribute values. The splitting process continues until the nodes become homogeneous (contain elements of one class) or they cannot be further divided in a meaningful way. Decision trees are easy to interpret, but can be prone to overfitting, which can be mitigated by pruning the tree.

3.2.2. Customizable parameters of the Decision Tree algorithm

Our Decision Tree implementation has two parameters provided when creating the tree.The first one is max_depth, which limits the maximum number of nodes that our tree can reach.Implementing this parameter allows you to influence accuracy by preventing overfitting. The second parameter is min_leaf which limits the minimum amount of information required to create another node. Implementing this parameter allows us to avoid a situation in which a single value causes the prediction to be incorrect.

3.2.3. Implementation of the algorithm

Our implementation contains the following functions: __init__(), train(), _create_tree(), _find_label_prob(), _find_best_split(), _sets_entropy(), _entropy(), predict(), _predict_one_case(). Below are descriptions of the following functions.

- __init__()
  - This function is run when creating the Tree and sets the parameters and required variables.
- train()
  - This function is responsible for starting the tree training and running the _create_tree function.
- _create_tree()
  - This function it is the main function responsible for building the tree that it checks whether the maximum depth has not been reached, if not using the _find_best_split function finds the best split of the available data. Then, using the TreeNode class, the data obtained from subsetting, and the _find_label_prob function, it creates

a new node. For both subsets, the condition of the minimum number of records is checked. If the condition is not met, the node is returned as a terminal node, otherwise, for the right and left branches of the node, the _create_tree function is called with increased depth for the appropriate sets.

- _find_label_prob()
  - This function returns the probability of occurrence of each class in the set.
- _find_best_split()
  - This function returns two sets with the lowest common entropy calculated by the sets entropy function, the function checks the entropy for three values in each feature. the function additionally returns the feature and the value for which the minimum entropy was obtained.
- _sets_entropy()
  - This function a returns the sum of the entropy calculated by the _entropy function multiplied by the share of the subset in the entire set.
- _entropy()
  - This function returns the sum of the results:

    $$-p \cdot \log_2(p)$$

    where p is the number of occurrences of the class in the set divided by the length of the set.
- predict()
  - This function for each row in the test set, it calls the _predict_one_case function and returns a list of all predicted classes.
- _predict_one_case()
  - This function compares the value for a given feature with the value stored in the node and, if possible, moves to the next node. If the next node does not exist, it returns the probabilities from the node.

3.3. Random Forest

3.3.1. Quick description of the Random Forest algorithm

Random forest is a machine learning algorithm that consists of multiple decision trees built on random subsets of training data. Each tree is trained independently, and the final prediction is the result of the vote.By randomly sampling data and randomly selecting features, random forest is resistant to overfitting and is suitable for many machine learning applications.

3.3.2. Customizable parameters of the Random Forest algorithm

Our Decision Tree implementation has four parameters provided when creating the forest. The first two max_depth and min_leaf are Decision Tree parameters. The two remaining parameters are tree_numb and sample_size. The first one is responsible for the number of trees that were created for the forest. The second one changes the amount of information that is sent to each tree.

3.3.3. Implementation of the algorithm

Our implementation contains the following functions: _init_(), train(), predict(). Below are descriptions of the following functions.

- __init__()
  - This function is run when creating the forest and sets the parameters and required variables.
- train()
  - This function creates the appropriate number of random trees and assigns data to them.
- predict()
  - This function calls the _predict function for each tree and then selects the most frequent answer for each row.

4. Adjusting the algorithm parameters

Changing the parameters of decision-making algorithms may have a significant impact on their effectiveness, which is why we conducted tests to find the best parameters.

4.1. Decision Tree

The Decision Tree has only two parameters, max_depth and mini_leaf. Below is a table of accuracy when changing these parameters.

The table above shows the accuracy in percentages for each parameter setting.On the horizon- tal axis we have the min_leaf parameter and on the vertical axis max_depth. The best results are obtained for depths greater than 10 and the number of leaves equal to 2. The obtained accuracy is 94.32%.For comparison, the classifier from the sklearn library achieved 90.69%, which gives us a better result by almost 4%.

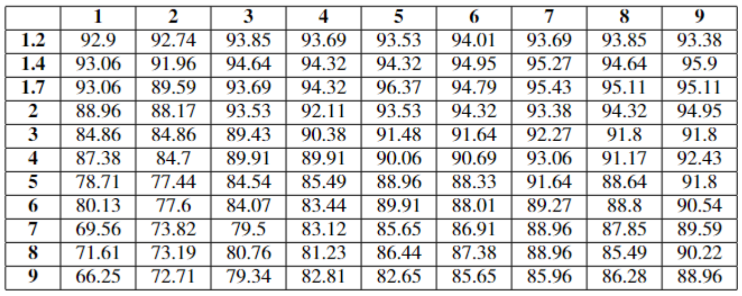
4.2. Random Forest

Since the Random Forest has four parameters, i.e. max_depth, min_leaf, tree_numb and sam- ple_size, we decided to conduct research only for the four best results obtained by the Decision Tree. The parameters for the following tables are respectively: (12,2),(12,4),(10,2),(10,4) where the first number is max_depth and the second is min_leaf.

Table 10
Accuracy table

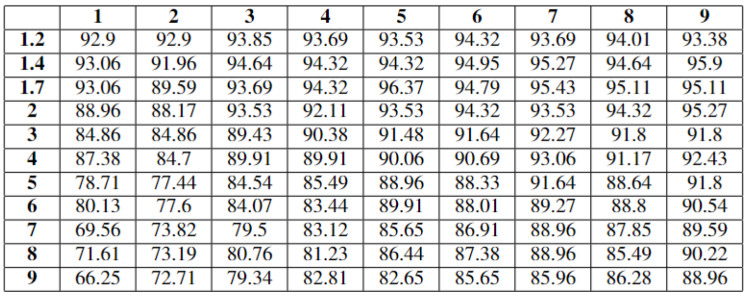
/	1	2	3	4	5	6	7	8
1	16.09	16.09	16.09	16.09	16.09	16.09	16.09	16.09
2	29.97	29.97	29.97	29.97	29.97	29.97	29.97	29.97
3	51.10	51.10	51.10	51.10	51.10	51.10	51.10	51.10
4	59.94	59.94	59.94	59.94	59.94	59.94	59.94	59.94
5	69.56	69.56	69.56	69.56	69.40	69.40	69.40	69.40
6	82.02	82.02	82.02	81.86	81.70	81.70	81.70	81.70
7	89.91	90.06	90.06	89.91	89.59	89.59	89.27	89.27
8	90.22	90.38	90.69	90.54	90.22	90.06	89.75	89.91
9	92.74	93.06	93.22	93.06	92.43	92.27	91.96	92.11
10	93.85	94.16	94.16	94.01	93.38	93.22	92.90	92.11
11	93.06	94.32	93.85	94.01	93.38	93.22	92.90	92.11
12	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
13	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
14	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
15	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
16	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
17	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11
18	93.38	94.32	93.85	94.01	93.38	93.22	92.90	92.11

TABLE XI
ACCURACY TABLE MAX_DEPTH = 12 MIN_LEAF = 2



	1	2	3	4	5	6	7	8	9
1.2	92.9	92.74	93.85	93.69	93.53	94.01	93.69	93.85	93.38
1.4	93.06	91.96	94.64	94.32	94.32	94.95	95.27	94.64	95.9
1.7	93.06	89.59	93.69	94.32	96.37	94.79	95.43	95.11	95.11
2	88.96	88.17	93.53	92.11	93.53	94.32	93.38	94.32	94.95
3	84.86	84.86	89.43	90.38	91.48	91.64	92.27	91.8	91.8
4	87.38	84.7	89.91	89.91	90.06	90.69	93.06	91.17	92.43
5	78.71	77.44	84.54	85.49	88.96	88.33	91.64	88.64	91.8
6	80.13	77.6	84.07	83.44	89.91	88.01	89.27	88.8	90.54
7	69.56	73.82	79.5	83.12	85.65	86.91	88.96	87.85	89.59
8	71.61	73.19	80.76	81.23	86.44	87.38	88.96	85.49	90.22
9	66.25	72.71	79.34	82.81	82.65	85.65	85.96	86.28	88.96

TABLE XII
ACCURACY TABLE MAX_DEPTH = 10 MIN_LEAF = 2



	1	2	3	4	5	6	7	8	9
1.2	92.9	92.9	93.85	93.69	93.53	94.32	93.69	94.01	93.38
1.4	93.06	91.96	94.64	94.32	94.32	94.95	95.27	94.64	95.9
1.7	93.06	89.59	93.69	94.32	96.37	94.79	95.43	95.11	95.11
2	88.96	88.17	93.53	92.11	93.53	94.32	93.53	94.32	95.27
3	84.86	84.86	89.43	90.38	91.48	91.64	92.27	91.8	91.8
4	87.38	84.7	89.91	89.91	90.06	90.69	93.06	91.17	92.43
5	78.71	77.44	84.54	85.49	88.96	88.33	91.64	88.64	91.8
6	80.13	77.6	84.07	83.44	89.91	88.01	89.27	88.8	90.54
7	69.56	73.82	79.5	83.12	85.65	86.91	88.96	87.85	89.59
8	71.61	73.19	80.76	81.23	86.44	87.38	88.96	85.49	90.22
9	66.25	72.71	79.34	82.81	82.65	85.65	85.96	86.28	88.96

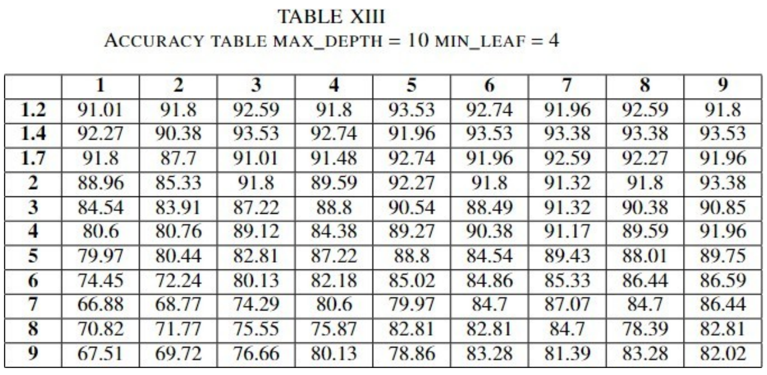


TABLE XIII
ACCURACY TABLE MAX_DEPTH = 10 MIN_LEAF = 4

	1	2	3	4	5	6	7	8	9
1.2	91.01	91.8	92.59	91.8	93.53	92.74	91.96	92.59	91.8
1.4	92.27	90.38	93.53	92.74	91.96	93.53	93.38	93.38	93.53
1.7	91.8	87.7	91.01	91.48	92.74	91.96	92.59	92.27	91.96
2	88.96	85.33	91.8	89.59	92.27	91.8	91.32	91.8	93.38
3	84.54	83.91	87.22	88.8	90.54	88.49	91.32	90.38	90.85
4	80.6	80.76	89.12	84.38	89.27	90.38	91.17	89.59	91.96
5	79.97	80.44	82.81	87.22	88.8	84.54	89.43	88.01	89.75
6	74.45	72.24	80.13	82.18	85.02	84.86	85.33	86.44	86.59
7	66.88	68.77	74.29	80.6	79.97	84.7	87.07	84.7	86.44
8	70.82	71.77	75.55	75.87	82.81	82.81	84.7	78.39	82.81
9	67.51	69.72	76.66	80.13	78.86	83.28	81.39	83.28	82.02

The table above shows the accuracy in percentages for each parameter setting.On the horizontal axis we have the min leaf parameter and on the vertical axis max depth. The best results are obtained for depths equal to 12, the number of leaves equal to 2 tree number equal to 5 and sample size equal to 1.7. The obtained accuracy is 96.37%.

5. Experiments

5.1. Remove one column

Below we present an experiment testing the change in precision when one of the columns is removed. This correlation matrix shows the relationships that exist between individual variables. This graph shows that if the FAVC columns were removed, an accuracy of 94.795 % would be achieved.

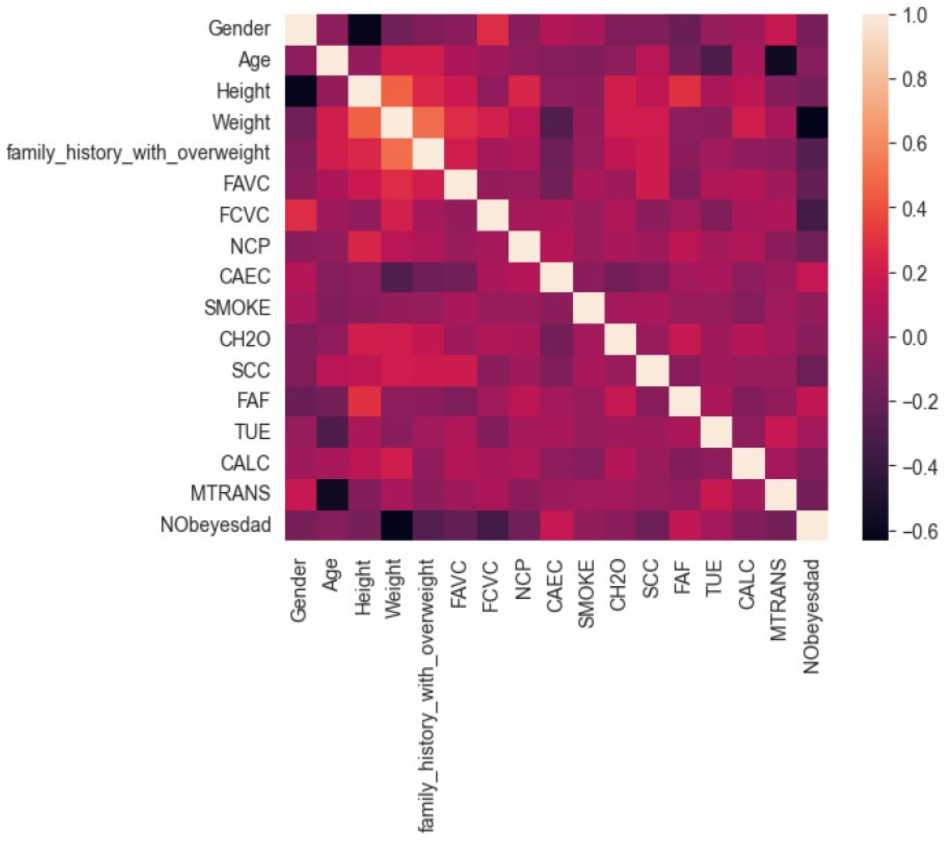


Figure 10: Correlation matrix

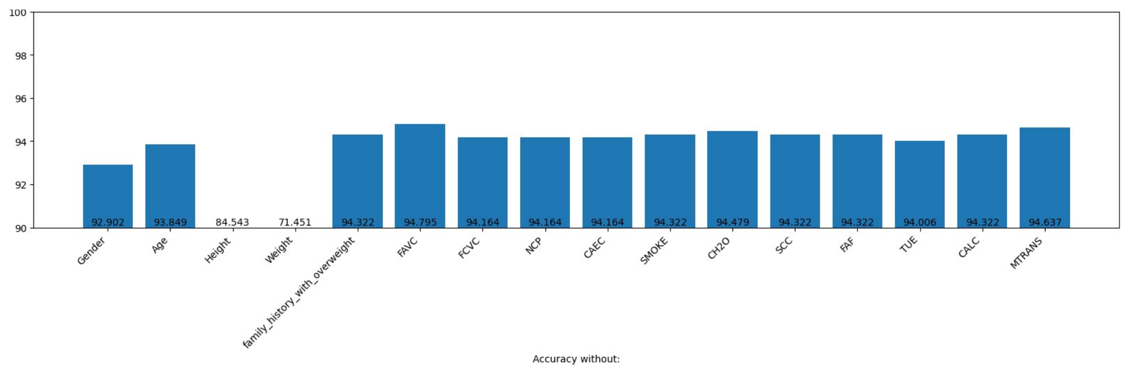


Figure 11: Accuracy without one column

Summary

In conclusion, the assumption that the algorithms that worked greatly in datasets related to childhood obesity, also gave high results in accuracy for our data, was proven to be correct. The exact estimates are 96.37% on Random forest and 94.32% on Decision Tree, which are both better results than the algorithms proposed by sklearn library. We also deduced that using them might be beneficial in making medical diagnosis and self-management of obesity.

References

[1] Shu-Zhong Jiang, Wen Lu, Xue-Feng Zong, Hong-Yun Ruan, Yi Liu, "Obesity and hyper- tension", Experimental and Therapeutic Medicine Volume 12 Issue 4, October 2016

[2] Bloomgarden ZT, "Third Annual World Congress on the Insulin Resistance Syndrome: associated conditions", Diabetes Care, 2006 Sep;29(9):2165-74, doi: 10.2337/dc06-zb09. PMID: 16936171

[3] Strumpf E: The obesity epidemic in the United States: causes and extent, risks and solutions. The Commonwealth Fund; New York, NY: 2004

[4] M. Woźniak, J. Siłka, i M. Wieczorek, „Deep neural network correlation learning mechanism for CT brain tumor detection”, Neural Computing Applications, t. 35, s. 14611–14626, 2023, doi: 10.1007/s00521-021-05841-x.

[5] J. Chaki i M. Woźniak, „Deep learning for neurodegenerative disorder (2016 to 2022): a systematic review”, Biomedical Signal Processing and Control, t. 80. Pt. 1, s. 1–20, 2023, doi: 10.1016/j.bspc.2022.104223.

[6] J. Chaki i M. Woźniak, „A deep learning based four-fold approach to classify brain MRI: BTSCNet”, Biomedical Signal Processing and Control, t. 85, s. 1–20, 2023, doi: 10.1016/j.bspc.2023.104902.

[7] W. Siłka, M. Wieczorek, J. Siłka, and M. Woźniak, “Malaria detection using advanced deep learning architecture,” Sensors, vol. 23, Art. no. 3, 2023, doi: 10.3390/s23031501.

[8] G. Colmenarejo, “Machine Learning Models to Predict Childhood and Adolescent Obesity:

[9] A Review.” Nutrients vol. 12,8 2466. 16 Aug. 2020, doi:10.3390/nu12082466

[10]A. Triantafyllidis, E. Polychronidou, A. Alexiadis, C. L. Rocha, D. N. Oliveira, A. da Silva,

[11] A. L. Freire, C. Macedo, I. F. Sousa, E. Werbet, E. A. Lillo, H. G. Luengo, M. T. Ellacuría, K. Votis, D. Tzovaras, “Computerized decision support and machine learning applications for the prevention and treatment of childhood obesity: A systematic review of the literature”, Artificial Intelligence in Medicine, Volume 104, 2020, 101844, ISSN 0933-3657,

[12] T. M. Dugan , S. Mukhopadhyay , A. Carroll , S. Downs, “Machine Learning Techniques for Prediction of Early Childhood Obesity”, Appl Clin Inform, 2015, 2015 Aug

[13] 12. doi:10.4338/ACI-2015-03-RA-0036

[14] J. Ali, Khan, K. Rehanullah, N. Ahmad, I. Maqsood, ”Random Forests and Decision Trees”, International Journal of Computer Science Issues(IJCSI) (2012) 9.

[15] D. J. Wu, T. Feng, M. Naehrig, K. Lauter, “Privately Evaluating Decision Trees and Random Forests”, Cryptology ePrint Archive, Paper 2015/386

[16] L. Breiman. Random forests. Machine Learning, 45(1):5–32, 2001.