BOOSTING ALGORITHMS APPROACH FOR DIABETES CARE

Abstract

In this project, the dataset that includes people with and without diabetes will first be edited for machine learning. Four different boosting algorithms will be trained on the dataset for diabetes care, and how well the algorithms perform will be examined.

Alper Birinci

November 25, 2022

Contents

Introduction

Diabetes is one of the most serious diseases in the world. It is estimated that approximately 415 million people have diabetes. According to the information shared by the World Health Organization, 1.5 million people died directly due to diabetes in 2019 (World Health Organization, 2022). Another data showing the importance of diabetes is that 48% of the people who die are younger than 70 years old (Diabetes.co.uk, 2019). It is expected that 643 million people by 2030 and 783 million people by 2045 will have diabetes. 966 billion dollars (9% of total spending on adults) in healthcare expenditures in 2021 is due to diabetes. (International Diabetes Federation, 2021)

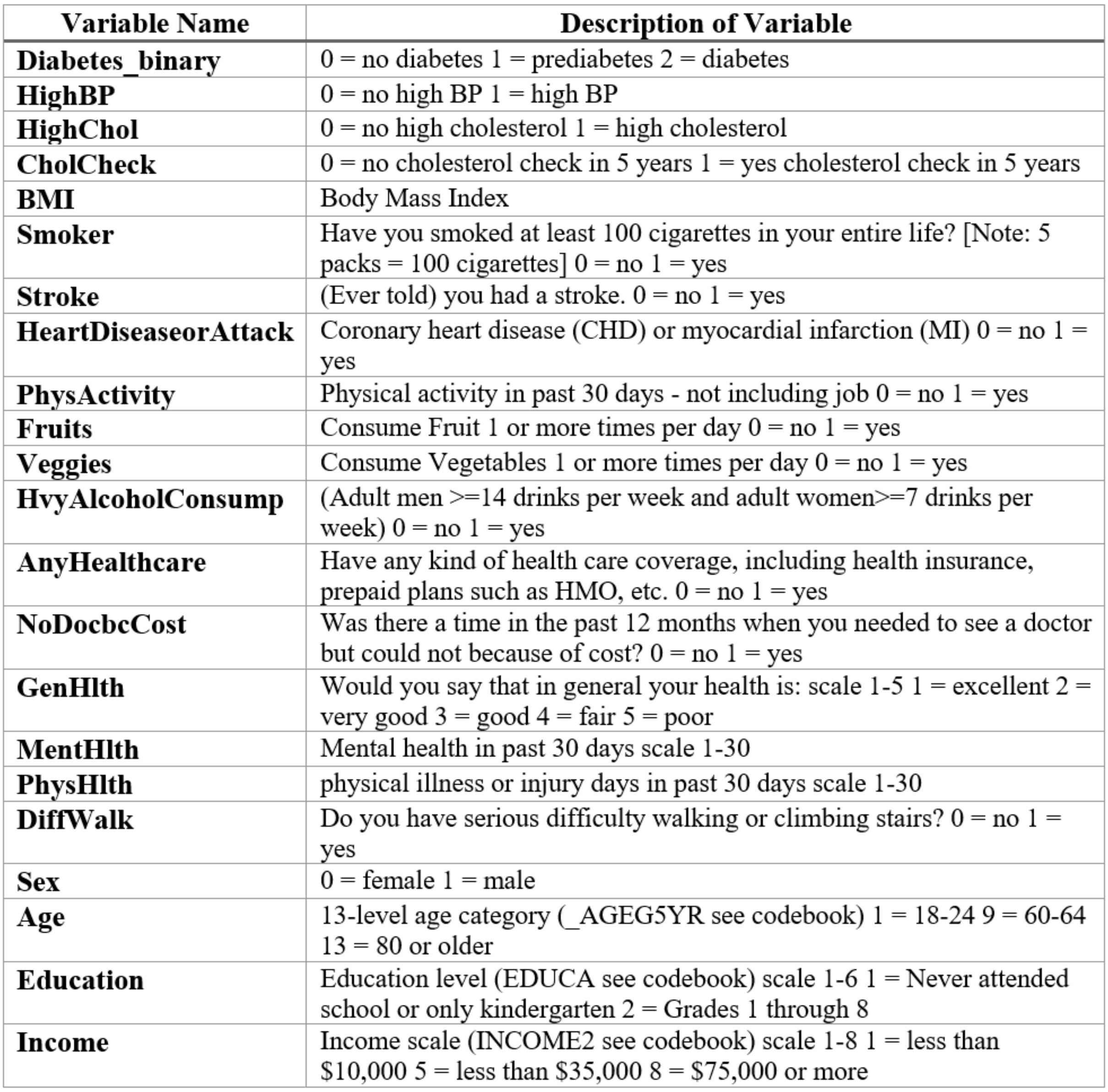
Unfortunately, people do not pay attention to diabetes, which is one of the biggest health problems in the world. If you don't know whether you have diabetes or not, the chances of your answer being yes are high. In middle- and low-income countries, 3 out of 4 adults have diabetes. Moreover, 1 in 2 adult diabetics are undiagnosed (International Diabetes Federation, 2021). This data shows that the awareness of diabetes in society should be increased. Diagnosing diabetes early and starting the treatment process early is the most important thing to prevent possible health problems. A person with diabetes may have health problems in his/her heart, blood vessels, nervous system, kidneys, and even eyes in the future. Therefore, people with diabetes need to establish order in their lives in order to be healthy individuals after their diagnosis such as weight management, diet program, daily exercises, and starting to use certain medications. With regular diabetes tests and people diagnosed with diabetes, starting a regular life and treatment process, the number of deaths due to diabetes in the world can be significantly reduced or they can live a healthier/more comfortable life.

The popularity of artificial intelligence is increasing day by day, it is very effective and used for many problems. However, AI still hasn't had a big impact on diabetes health. Artificial intelligence can make a big difference in processing big data and early diagnosis of people's health problems. With the support of the government, general health checks carried out in hospitals, and the information obtained about individuals stored as complete (without missing) data, artificial intelligence can be developed for people to lead healthier lives.

Background

In this project, 4 different machine learning algorithms will be developed on a complete (without missing data) dataset available on the "Kaggle" site. The names of these 4 algorithms are Gradient Boosting Machines (GBM), eXtreme Gradient Boosting (XGBoost), Light Gradient Boosting Machines (LightGBM), and Catboost. 4 Algorithm has been developed to train on the presented dataset and then make predictions according to this training. When we consider the patient data in a hospital and the size of this dataset, it will not be enough to compare the performance of the algorithms with the accuracy rates alone. The main purpose of this project is to determine the most suitable algorithm for large datasets by measuring both the speed and accuracy of all 4 boosting algorithms.

The "Diabetes Health Indicators Dataset (BRFSS 2015)" dataset to be used in the project is publicly shared on the Kaggle site (Teboul, 2022). The shared dataset is the cleaned-up version of the BRFSS 2015 dataset and is optimized for machine learning. In other words, there is no need to perform cleaning operations on the dataset because there is no missing value. However, due to the imbalance in the number of target values, scaling operations are required. There are 22 attributes in the dataset, 253680 examples, 1st result, and 2nd to 22nd column features. You can find the properties of the attributes shared in the dataset in ***table 1***.



Variable Name	Description of Variable
Diabetes_binary	0 = no diabetes 1 = prediabetes 2 = diabetes
HighBP	0 = no high BP 1 = high BP
HighChol	0 = no high cholesterol 1 = high cholesterol
CholCheck	0 = no cholesterol check in 5 years 1 = yes cholesterol check in 5 years
BMI	Body Mass Index
Smoker	Have you smoked at least 100 cigarettes in your entire life? [Note: 5 packs = 100 cigarettes] 0 = no 1 = yes
Stroke	(Ever told) you had a stroke. 0 = no 1 = yes
HeartDiseaseorAttack	Coronary heart disease (CHD) or myocardial infarction (MI) 0 = no 1 = yes
PhysActivity	Physical activity in past 30 days - not including job 0 = no 1 = yes
Fruits	Consume Fruit 1 or more times per day 0 = no 1 = yes
Veggies	Consume Vegetables 1 or more times per day 0 = no 1 = yes
HvyAlcoholConsump	(Adult men >=14 drinks per week and adult women>=7 drinks per week) 0 = no 1 = yes
AnyHealthcare	Have any kind of health care coverage, including health insurance, prepaid plans such as HMO, etc. 0 = no 1 = yes
NoDocbcCost	Was there a time in the past 12 months when you needed to see a doctor but could not because of cost? 0 = no 1 = yes
GenHlth	Would you say that in general your health is: scale 1-5 1 = excellent 2 = very good 3 = good 4 = fair 5 = poor
MentHlth	Mental health in past 30 days scale 1-30
PhysHlth	physical illness or injury days in past 30 days scale 1-30
DiffWalk	Do you have serious difficulty walking or climbing stairs? 0 = no 1 = yes
Sex	0 = female 1 = male
Age	13-level age category (_AGEG5YR see codebook) 1 = 18-24 9 = 60-64 13 = 80 or older
Education	Education level (EDUCA see codebook) scale 1-6 1 = Never attended school or only kindergarten 2 = Grades 1 through 8
Income	Income scale (INCOME2 see codebook) scale 1-8 1 = less than $10,000 5 = less than $35,000 8 = $75,000 or more

Table 1: Attributes and descriptions of values in the dataset (Teboul, 2022)

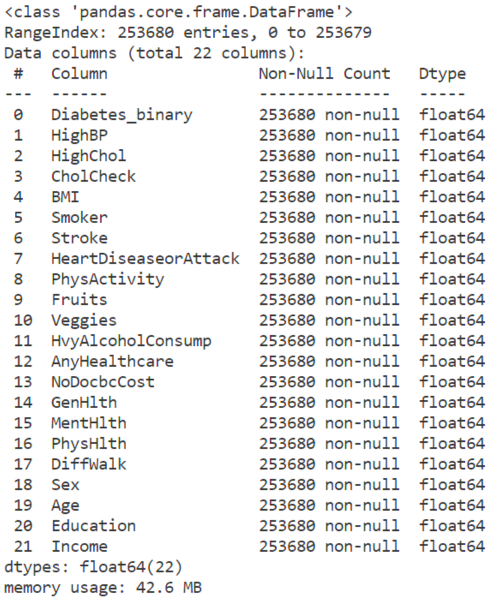
Python will be used throughout this project. It has nice libraries that are helpful for dataset operations and machine learning. The "matplotlib.pyplot" library will generally be used for graphics and visual representation. Thanks to the "time" library, how long the algorithms work will be measured. The "pandas" library will be used to read the dataset and then import it into python properly. The "imblearn" library will be used for the SMOTE algorithm. "sklearn", "xgboost", "catboost" algorithms will be the libraries to be used for machine learning.

Methodology and Data

In this section, in order to measure the performance of the algorithms in the best way, the data will be checked first and if necessary, the data will be edited. The performance of 4 algorithms will be measured with the python implementation. Afterward, boosting algorithms will be run with test rates of 0.25, 0.30, and 0.40, respectively. Obtained results will be shown visually and numerically with tables.

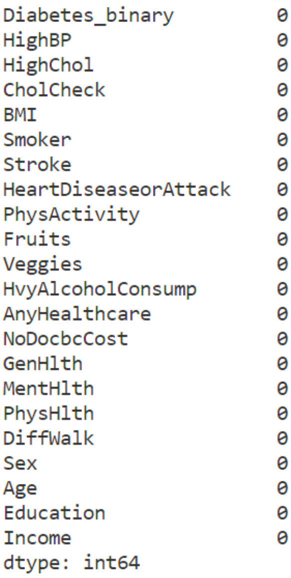
Data Control and Regulation

At first, the dataset will be checked for missing data. If attributes with too much missing data are found, those attributes will be removed from the dataset. In ***Image 1***, df.info() is used to get information about the attributes. ***Image 2*** has the output of the code used to see how many data are missing in each column.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 253680 entries, 0 to 253679
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   Diabetes_binary       253680 non-null  float64
 1   HighBP                253680 non-null  float64
 2   HighChol              253680 non-null  float64
 3   CholCheck             253680 non-null  float64
 4   BMI                   253680 non-null  float64
 5   Smoker                253680 non-null  float64
 6   Stroke                253680 non-null  float64
 7   HeartDiseaseorAttack  253680 non-null  float64
 8   PhysActivity          253680 non-null  float64
 9   Fruits                253680 non-null  float64
 10  Veggies               253680 non-null  float64
 11  HvyAlcoholConsump     253680 non-null  float64
 12  AnyHealthcare         253680 non-null  float64
 13  NoDocbcCost           253680 non-null  float64
 14  GenHlth               253680 non-null  float64
 15  MentHlth              253680 non-null  float64
 16  PhysHlth              253680 non-null  float64
 17  DiffWalk              253680 non-null  float64
 18  Sex                   253680 non-null  float64
 19  Age                   253680 non-null  float64
 20  Education             253680 non-null  float64
 21  Income                253680 non-null  float64
dtypes: float64(22)
memory usage: 42.6 MB
```

Image 1: Information about the columns



```
Diabetes_binary         0
HighBP                  0
HighChol                0
CholCheck               0
BMI                     0
Smoker                  0
Stroke                  0
HeartDiseaseorAttack    0
PhysActivity            0
Fruits                  0
Veggies                 0
HvyAlcoholConsump       0
AnyHealthcare           0
NoDocbcCost             0
GenHlth                 0
MentHlth                0
PhysHlth                0
DiffWalk                0
Sex                     0
Age                     0
Education               0
Income                  0
dtype: int64
```

Image 2: Sum of missing data for each column

It has been observed that there is no missing data in the dataset. Therefore, there is no need to remove any attributes.

The sequence will be checked for duplicate data and the duplicate data will be removed from the table. The table has 24206 duplicates of data. Duplicate data is removed from the table with “drop_duplicates()”, leaving 229474 instances.

In the next part of the data editing, the imbalance in the target column needs to be optimized for machine learning. The difference between the number of people with diabetes and the number of people without diabetes is huge. In the data table, 194377 people do not have diabetes, while 35097 people have diabetes (***Table 2***). The imbalance can be understood more easily with the percentage pie (***Figure 1***).

Number of people without diabetes (0)	194377
Number of people with diabetes (1)	35097

Table 2: Numerical representation of target column's values

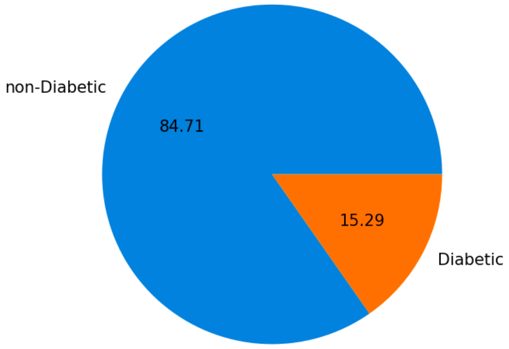


Figure 1: Percentage of target column's value

SMOTE (Synthetic Minority Over-Sampling Technique) will be used to correct the imbalance. SMOTE is one of the most common methods used to solve the imbalance problem. It aims to balance the class distribution by increasing the examples that fall into the minority class. SMOTE Creates new instances in the minority class. In the Smote algorithm, the Euclidean distance between the examined feature and its nearest neighbor is measured. It is then multiplied by a random number between 0 and 1 and then added to the examined attribute to create a new sample. For the implementation of SMOTE, the ready-made function in the "imblearn" library will be used (GeeksforGeeks, 2019).

Methodology

The 4 algorithms written below will be trained together with the dataset after the above operations. Both visual and numerical results will be shared in the analysis section according to the running time and accuracy of all algorithms one by one. Certain python libraries will be used for all algorithms, which simplifies implementation.

Boosting

Boosting is a machine-learning method that enables multiple decision trees to be implemented in a dataset. New trees are created based on errors (make a weak learner a strong learner). The most common example when describing boosting is golf. According to various factors, the decision tree estimates that it can cover 70 meters of 100 meters in 1 hit, and the process ends here. On the other hand, boosting algorithms aim to achieve better results by using decision trees many times.

GBM (Gradient Boosting Machine)

The method consists of the combination of the words gradient descent and boosting. GBM aims to achieve better results by constructing new trees according to the errors in previous decision trees using the gradient descent algorithm.

XGBoost (eXtreme Gradient Boosting)

XGBoost is an algorithm that emerged with the article "XGBoost: A Scalable Tree Boosting System" (Chen and Guestrin, 2016). It is an optimized version of the GBM algorithm. It is faster and has a better prediction rate than DBM. One of the best aspects of this algorithm is that the algorithm can handle over-learning on its own.

LightGBM (Light Gradient Boosting Machine)

It is a boosting algorithm developed as a Microsoft DMTK project in 2017. It is introduced in the article “LightGBM: A Highly Efficient Gradient Boosting Decision Tree” (Ke et al., 2017). According to the analysis made in the article, LightGBM is 20 times faster than other models. This is the biggest reason why this algorithm is used for big data. So much so that it is one of the most preferred machine learning algorithms today. Features of the algorithm include parallel learning, less RAM usage, and GPU support.

CatBoost

It is a machine-learning algorithm developed by Yandex in April 2017. It was introduced in the article “CatBoost: unbiased boosting with categorical features” (Prokhorenkova et al., 2017). The algorithm can be trained with categorical, numerical, or text data. The algorithm also has features such as visualization and GPU support.

Analysis and Discussions

All 4 algorithms will be run on the structured data structure. 3 different training test rates will be evaluated. Performances will be measured in duration and accuracy during each test. For each test rate, 10 trials will be performed and then averaged. All results will be displayed in the tables (***Table 3, Table 4, Table 5***).

75:25 Ratio

	GBM		**XGBoost**		**LightGBM**		**CatBoost**	
	Time	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**
Avg.	12.3	0.8516795	2.3	0.8507556	0.35	0.852185	13.87	0.8508254

Table 3: Average performance chart of 75:25 ratio

70:30 Ratio

	GBM		**XGBoost**		**LightGBM**		**CatBoost**	
	Time	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**
Avg.	11.57	0.8521128	2.08	0.8511686	0.33	0.8523016	12.87	0.8515608

Table 4: Average performance chart of 70:30 ratio

60:40 Ratio

	GBM		**XGBoost**		**LightGBM**		**CatBoost**	
	Time	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**	**Time**	**Accuracy**
Avg.	9.43	0.8522388	1.87	0.8505393	0.28	0.8521081	11.33	0.851291

Table 5: Average performance chart of 60:40 ratio

According to the tests, LightGBM was the fastest algorithm in all 3 ratios. LightGBM is about 10 times faster than XGBoost and more than 20 times faster than other algorithms. This proves how good LightGBM is in terms of speed compared to other algorithms. Although LightGBM is better in terms of accuracy except for the 60:40 ratio, the values obtained from all algorithms are good and almost identical to each other.

Conclusion and Future Work

In conclusion, boosting algorithms are machine learning algorithms that are easy to interpret, resistant to over-learning, and have a good prediction rate. The performances of the algorithms used in the project are very close to each other in terms of consistency. However, it is obvious that the LightGBM algorithm is many times faster than other algorithms. Nationwide data on any given disease would have very large samples with lots of value. In this case, programmers will want to prefer the speed performance that LightGBM offers them. With LightGBM, the rate of people diagnosing diabetes late can be significantly reduced.

If I had more time for the report, I would create a user interface with Python using the "tkinter" library. In this interface, I would create a program that predicts whether the patient has diabetes or not, according to the measurements of the last time he went to the hospital. I would discover what I need to do to use algorithms more effectively for this program to work more efficiently.

Bibliography

1. World Health Organization (2022). *Diabetes*. [online] World Health Organization. Available at: https://www.who.int/news-room/fact-sheets/detail/diabetes.

2. Diabetes.co.uk (2019). *How Many People Have Diabetes - Diabetes Prevalence Numbers*. [online] Diabetes.co.uk. Available at: https://www.diabetes.co.uk/diabetes-prevalence.html.

3. International Diabetes Federation (2021). *International Diabetes Federation - Facts & Figures*. [online] Idf.org. Available at: https://idf.org/aboutdiabetes/what-is-diabetes/facts-figures.html.

4. Teboul, A. (2022). *Diabetes Health Indicators Dataset*. [online] www.kaggle.com. Available at: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset.

5. GeeksforGeeks. (2019). *ML | Handling Imbalanced Data with SMOTE and Near Miss Algorithm in Python*. [online] Available at: https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/.

6. Chen, T. & Guestrin, C., 2016. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD &#x27;16. New York, NY, USA: ACM, pp. 785–794. Available at: http://doi.acm.org/10.1145/2939672.2939785.

7. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. [online] Available at: https://proceedings.neurips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf.

8. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. and Gulin, A. (2017). *CatBoost: unbiased boosting with categorical features*. [online] Available at: https://proceedings.neurips.cc/paper/2018/file/14491b756b3a51daac41c24863285549-Paper.pdf.