Diabetes Prediction using Machine Learning Algorithms

Nugara Andrea(903051)

Patrick Costa(858156)

Matteo Mondini(902873)

Index

1. Abstract

Diabetes Mellitus (DM) is a chronic metabolic disease that arises due to abnormally high levels of blood glucose, known as hyperglycemia. Timely diagnosis and treatment are crucial to mitigate life-threatening risks associated with this condition.Fortunately, advances in technology have enabled earlier detection of the disease.In this study, we have developed a decision support system using machine learning algorithms to aid in DM diagnosis. Our approach begins with a very detailed exploratory data analysis, followed by data pre-processing and feature selection techniques. Important features are selected, and the data is trained using four different machine learning algorithms, including Naïve Bayes, Random Forest, Gradient Boosting and Multilayer Perceptron. Performance metrics were used to evaluate the experimental results of the machine learning algorithms. Among these algorithms, Gradient Boosting exhibited the lowest Log Loss with a score of 0.49 in hold-out and 0.507 in a 5 folds cross-validation. This outcome demonstrates the efficacy of our approach and highlights the potential for machine learning algorithms to aid in the early diagnosis and treatment of DM.

2. Introduction

With the world population on the rise, there is a growing need to develop computational systems that can enhance healthcare outcomes. Ongoing studies in the healthcare sector are constantly evolving, leading to the development of highly efficient and reliable decision-making systems that improve accuracy in disease diagnosis.
It is crucial to use the capabilities of advanced methods and technologies, such as artificial intelligence and machine learning, to create new approaches and applications in the medical field.
Our proposed methodology is focused on a dataset and uses its data and features to train a machine learning algorithm to predict diabetes with capable metrics to compute values such as accuracy and recall.

3. Dataset

The dataset is constructed from 40108 records and 17 features variables and 1 target variable 'Diabetes' and This is a binary classification problem where 0 means no diabetes and 1 means diabetes. There are 13 binary features and 5 discrete variables.

The features are:

Age: 3-level age category 1 = 18-24, 9 = 60-64, 13 = 80 or older.
Sex: 0 = female, 1 = male.
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.
HeartDiseaseorAttack: Coronary heart disease (CHD) or myocardial infarction (MI) 0 = no, 1 = yes.
PhysActivity: Physical activity in the past 30 days - not including job 0 = no, 1 = yes.
Fruits: Consume Fruit one or more times per day 0 = no, 1 = yes.
Veggies: Consume Vegetables 1 or more times per day 0 = no, 1 = yes.
HvyAlcoholConsump: Adult male: more than 14 drinks per week. Adult female: more than 7 drinks per week. 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: Days of poor mental health scale 1-30 days.
PhysHlth: Physical illness or injury days in the past 30 days scale 1-30.
DiffWalk: Do you have serious difficulty walking or climbing stairs? 0 = no, 1 = yes.
Hypertension: 0 = no hypertension, 1 = hypertension.

4. Methodology

In this study, we followed a standard methodology for developing a machine learning model for predicting diabetes. The process involved several steps, including data preprocessing, transformation, feature selection, algorithm selection, and hyperparameter optimization. We also used partitioning and cross-validation to create a validation set and evaluate the performance of the model.

First, we started with the Diabetes dataset, which contained several features such as age, BMI,sex among others. We preprocessed the data to handle anomalies that could affect the performance of the model. Next, we transformed the data by applying various techniques such as normalization, and we tried to encode some variables.

After preprocessing and transformation, we applied feature selection techniques to identify the most relevant features for predicting diabetes. We used genetic algorithm and tournament selection to select the best features that had a significant impact on the performance of the model.

Next, we applied several machine learning algorithms to the selected features and evaluated their performance using different metrics such as accuracy, error rate, log loss, and Cohen's kappa. We used popular algorithms such as Naïve Bayes,, Random Forest, gradient boosting and Multilayer Perceptron. First, we partitioned our data into a training set and a validation set. We then used the training set to train the model with different hyperparameters. For each combination of hyperparameters, we calculated the log loss on the validation set using 5-fold cross-validation.We applied a range of hyperparameters to optimize our model. These included i.e. the number of trees in the random forest, the maximum depth of the trees, the learning rate of the gradient boosting algorithm.We used Bayesian optimization to find the optimal hyperparameters that yielded the best performance for each algorithm.

After hyperparameter optimization, we evaluated the performance of the model on the holdout set and using 5-fold cross- validation. The algorithm that performed the best was Gradient Boosting, with a log loss of
0.49 in the holdout set and 0.5078 in 5-fold cross-validation. These results suggest that our model is robust and can accurately predict the likelihood of diabetes in patients.

In conclusion, our methodology involved several steps, including data preprocessing, transformation, feature selection, algorithm selection, and hyperparameter optimization. We used partitioning and cross-validation techniques to evaluate the performance of the model.

5. Exploratory analysis

During the exploratory analysis, we tried to gain a better understanding of the composition of our dataset such as missing data, number of records, number of features and their distribution and whether these had a correlation with each other or whether some were useless. In the end, we found that there were no missing values and that no feature had a correlation with another greater than 80%, so they are independent of each other. The distribution along all records was balanced. The analysis continued by looking for any outliers which were found but were kept as the model performed better. We also built a DataApp on Knime with the univariate and bivariate analyses. In our data app we performed an in depth and interactive Exploratory Analysis of the dataset.

6.Preprocessing

For the pre-processing part, we transformed the target binary variable (Diabetes) into a string, so that it could be used in the learning phase; in addition, we normalized the variables that are not binary and have a high range of values, more precisely the variables 'Age', 'BMI', 'Mental Health', 'Physical Health'. The aim is to bring the different input characteristics to the same order of magnitude so that they can be compared equally across all variables and influence the model in a similar way.
We also tried to binarize all the variables that were not, thus consistently increasing the number of features, the goal was to make the model more efficient: the binarization of the variables in fact can reduce the number of operations necessary to calculate the predictions of the model. This results in a more efficient and faster model, but since we didn't see any performance improvement from the classifier, we abandoned this path.
We also tried the features engineering by aggregating the columns that had a logical and conceptual similarity, for example by creating a health index, built through the following formula: (MentHlth + PhysHlth +GenHlth)/3. Again, no improvements were found.

7. Machine learning algorithms

In this work, several classification techniques in the field of Machine Learning have been applied, in particular we used the following:

Naïve Bayes: Naive Bayes (NB) is a classification approach derived from Bayes theorem. The term "naive" refers to the two assumptions made by the technique: the first being that predictive features are independent, and the second being that there are no hidden features. The Naive Bayes classification technique calculates the probability of various classes.To summarize, NB is a classification approach that applies Bayes theorem, where it is assumed that the predictive features are independent and that there are no hidden features. The technique calculates the probability of different classes, thus making it a useful tool for classification problems.

Random Forest: Random Forest is an ensemble technique that belongs to the "Bagging" family and is used for both classification and regression problems. It is based on the idea of combining multiple decision trees in order to improve their performance. The algorithm works by creating a set of decision trees, each of which is trained on a random subset of the data.During the creation of the decision trees, a random subset of the features is selected at each node to split the data. This results in a diverse set of decision trees that are less prone to overfitting, as the trees are built on different subsets of the data and features.When a new data point is to be classified, the Random Forest algorithm aggregates the predictions of all the decision trees and selects the class that receives the most votes. The use of an ensemble of decision trees makes the model more robust and accurate compared to individual decision trees.

Gradient Boosting: Provides a prediction model that is formed by an ensemble of weak prediction models, typically decision

trees.When decision trees are used as the weak learner, the resulting algorithm is called gradient-boosted trees. This algorithm generally outperforms the popular random forest technique.Gradient-boosted trees models are built in a stage-wise manner similar to other boosting methods. However, this approach is generalized by allowing optimization of any differentiable loss function. This flexibility in optimization enables the model to be fine-tuned to specific data sets and can lead to better performance compared to other boosting methods.
Multilayer Perceptron: Multilayer Perceptron (MLP) is a neural network that uses multiple layers of interconnected nodes to process input data and generate output. MLP is trained using backpropagation and has been successfully used in applications such as speech recognition and image classification.

8. Boosting

Boosting is an ensemble learning technique that combines multiple weak learning algorithms to create a stronger prediction model. Boosting works by iteratively training multiple models on the same data, with eachsubsequent model focusing on the samples that were misclassified by the previous model. By combining the predictions of theseindividual models, the boosting algorithm is able to produce a highly accurate prediction.

One approach to boosting is to use the Naive Bayes algorithm as the weak learning algorithm. Naive Bayes is a simple but effective algorithm that works well with small and high-dimensional datasets. By using Naive Bayes in the boosting algorithm, we can improve the performance of the model byreducing the risk of overfitting and increasing the model's generalization ability. In addition,Naive Bayes is computationally efficient, making it a good choice for large datasets.
The optimized algorithm, after an ad-hoc feature selection archived a Logloss score of 0.75 in a 5 folds cross validation. After the boosting, the performances increased a lot,and it achieved a Logloss score of 0.56. In our case the iterations used were 10. We have therefore demonstrated how boosting the performance of the model can besignificantly improved.

9. Hyperparameter optimization

In the field of machine learning, hyperparameter optimization, also referred to as hyperparameter tuning, is a significant challenge that involves selecting an optimal set of hyperparameters for a given learning algorithm. Hyperparameters are parameters that govern the learning process and control the model's behavior. Unlike other parameters such as node weights, hyperparameters are not learned during training but are rather set by the user prior to the training phase.It is essential to select appropriate hyperparameters for a machine learning model to obtain the best possible results. Different data patterns may require different hyperparameters, such as constraints, weights, and learning rates, to generalize effectively. If the hyperparameters are not tuned correctly, the model's performance may be suboptimal, leading to a failure in solving the machine learning problem effectively.For model optimization weused the Bayesian optimization, it involves constructing a probabilistic model of the objective function, which is updated using Bayesian inference. The model is used to select the next set of hyperparameters to evaluate, based on an acquisition function that balances exploration and exploitation. Byiteratively updating the model and selecting the next hyperparameters to evaluate, Bayesian optimization can converge to the optimal set of hyperparameters efficiently.

10. Feature selection

Feature selection is a crucial step in machine learning that involves identifying and selecting the most relevant and informative features from a given dataset for building a predictive model. The goal of feature selection is to improve the model's performance, reduce overfitting, and enhance interpretability by removing irrelevant or redundant features.

Genetic algorithm is a popular technique for feature selection that is inspired by the process of natural selection. In genetic algorithm-based feature selection, a population of candidate feature subsets is created, and each subset is evaluated using a fitness function that measures its quality based on the model's performance. The fittest subsets are then selected for reproduction, where new subsets are created by combining the features of the selected subsets through crossover and mutation operations. The new subsets are then evaluated, and the process continues until a stopping criterion is met.In our case we decided to stop the algorithm after 20 generations without improvements.We decided to use the tournament selection criteria, In this method, a subset of individualsis randomly selected from the population, andthe one with the highest fitness value is chosen as the winner. The winner then undergoes crossover and mutation to create new offspring, which are added to the population. Tournament selection helps to maintain diversity in the population and prevent premature convergence to a suboptimal solution. It is a simple and effective method for selecting the fittest individuals in the population.Ad-hoc features have been selected for each machine learning algorithm so that each performs at itsoptimal level. Below there is a list of features that have been excluded for each algorithm used:

Naïve Bayes:”sex”, “healthdiseaseorattack”,”physactivity”,”fruits”,” menthlth”, “physhlth”
Random Forest: ”Smoker”,”Stroke”,Menthlth”
Gradient Boosting:”veggies”, “physhlth”
Multilayer Perceptron: ”sex”,”smoker”,”veggies”, “Menthlth”, “physhlth”, “stroke”
Performance evaluation

To evaluate the performance of machine learning algorithms, we categorize them based on their output and then observe certain parameters. There are several performance metrics that can be used to evaluate the accuracy and effectiveness ofthese algorithms.

Some of the most commonly used performance metrics include:

1. Accuracy: This measures how often the algorithm correctly predicts the outcome of a given task. It is calculated as the number of correct predictions divided by the total number of predictions.
2. Error: This measures the overall performance of the algorithm by calculating the difference between the predicted values and the actual values. Lower error values indicate better performance.
3. Logloss: This measures the accuracy of the predicted probabilities of a classification model. It is calculated as the negative log-likelihood of the predicted probabilities. In particular , in our case we implemented a python script to automatize the calculation with scikit-learn metric. The formula is the following:

$$L_{\log}(y, p) = -(y \log(p) + (1 - y) \log(1 - p))$$

4. Cohen's kappa: This measures the agreement between two annotators or between an annotator and a model in a classification task. It takes into account the possibility of agreement occurring by chance and provides a more accurate measure of agreement than simple percent agreement.

12.Results

In this section, the results of the projects will be presented. Two approaches will be used to evaluate the performance of the models. First, the models will be evaluated using a hold-out method. Second, a more robust evaluation will be performed using a 5-fold cross-validation method, which is considered a more robust approach for model evaluation.

12.1 Holdout results

Holdout is a simple method of dividing the available data into two parts,(in our case 80% for train and 20% for test). The model is trained on the training set, and the testing set is used to evaluate the model's performance. One of the main advantages of holdout is that it is relatively simple and easy to implement.

It requires minimal computational resources and is widely used in practice. However, one of the main disadvantages of holdout is that the evaluation may be sensitive to how the data is split. If the data is not representative of the overall population, the model's performance may be overestimated or underestimated.

	Logloss	Accuracy	Cohen's kappa	Error
Naïve Bayes	0.566	74.25%	0.484	25.75%
Random Forest	0.68	74.14%	0.502	25,86%
Gradient Boosting	0.498	75.78%	0.515	24,22%
Multilayer Perceptron	0.515	74,5%	0.489	25,5%

12.2. Cross validation results

Cross-validation, on the other hand, is a more sophisticated technique that involves dividing the data into multiple parts or "folds". The model is trained on a subset of the data and tested on the remaining parts, with the process repeated multiple times, each time with a different fold used for testing. This allows for a more robust evaluation of the model's performance and can help to reduce the variance in the evaluation. The disadvantage of cross-validation is that it can be computationally expensive, especially for large datasets.

	Logloss	Accuracy	Cohen's kappa	Error
Naïve Bayes	0.75	73,12%	0.461	26,79%
Random Forest	0.715	74.67%	0.492	25,33%
Gradient Boosting	0.5078	74,78%	0.494	25,22%
Multilayer Perceptron	0.513	75%	0.498	25%

13.Conclusions

In conclusion, after exploring various machine learning algorithms for the classification task and evaluating their performance, we found that the gradient boosting algorithm emerged as the top performer. The evaluation was conducted using several performance metrics, including accuracy, error, log loss, and Cohen's kappa. The evaluation process revealed that the gradient boosting algorithm achieved a log loss of 0.49 in the holdout dataset, indicating its ability to accurately classify the target variable with a high degree of precision. Furthermore, the algorithm also performed well in the 5-fold cross-validation, achieving a log loss of 0.5078, which provides further evidence that the model is robust and generalizable.

Overall, these results provide valuable insights into the effectiveness of machine learning algorithms for the classification task, and demonstrate the importance of performing a thorough evaluation of different models. Furthermore, these results indicate that the gradient boosting algorithm is a suitable choice for the classification of diabetes patients, and could be useful in developing predictive models that accurately classify patients and improve clinical decision-making.

REFERENCES

1. ***Yu, L., Liu, H. (2003). Feature selection forhigh dimensional data: A fast correlation-based filter solution. Proceedings of the Twentieth International Conference on Machine Learning, Washington DC***
2. ***Machine Learning: A ProbabilisticPerspective Kevin P. Murphy***
3. ***Yu, L., Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. Journal of Machine Learning Research, 5(1): 205-1224.***
4. ***https://scikit-learn.org/stable/modules/generated/sklearn.metrics.log_loss.html***
5. ***https://hub.knime.com/***
6. **Breiman, L. (2001). Random forests. Machine Learning, 45(1): 5-32. https://doi.org/10.1023/a:1010933404324**