Diabetes prediction classification model

Matteo Corona, Costanza Pagnin, Marta Pirovano

February 17, 2023

Abstract

The purpose of the project is to provide a model, trained on historical data, to classify new patients as either likely or unlikely to have diabetes. The work is made up of different stages. First of all we tested various models and chose the best one according to the lowest logloss score. We opted for a *Bayesian network* classificaition algorithm. Once the model was identified, to carry out the classification task we created a deployment workflow that takes a test dataset as input. The test dataset includes information on new patients, without the diabetes attribute. In the deployment workflow the previously trained model is applied to the test dataset to produce diabetes predictions. Finally, we prepared a web application to give the possibility to run the model and make predictions without interacting with the KNIME workflows.

1 Introduction

Diabetes is a chronic disease that affects millions of people worldwide and has become a major public health concern. It is a complex condition that requires accurate and timely diagnosis for effective treatment and management. Machine learning techniques have shown great promise in the field of medical diagnosis and have been successfully used in the classification of various diseases. In this report, we explore the application of machine learning algorithms for the classification of diabetes. Specifically, we will investigate the effectiveness of different classification models in accurately predicting the presence or absence of diabetes in patients based on their clinical information. The report will provide an overview of the dataset used, the machine learning techniques employed, and the performance metrics used to evaluate the models.

2 Data Exploration

The training dataset we have been provided with includes 18 attributes related to different aspects of a patient's health, including the target attribute, diabetes. We selected the most relevant characteristics for predicting diabetes and used them to train the best suited model for the task.
The dataset is composed of the following attributes:

- **age**: age categories, divided in bins from 1 to 13, with an average of a 5 year gap (1=18-24 y.o., 2=25-29 y.o., ... 12=75-79 y.o., 13=80 y.o.

or older)
Only in the data app (see section 6), for visual purposes, the variable age has been converted from bins to age values, represented by the corresponding bin midpoint.

- **sex**: 0=female, 1=male
- **HighChol**: 0=low cholesterol, 1=high cholesterol;
- **CholCheck**: 0=no cholesterol check in 5 years 1=cholesterol check in 5 years;
- **BMI**: Body Mass Index;
- **Smoker**: whether a person smoked more than 100 cigarettes, corresponding to 5 packs, in their entire life (0=no, 1=yes);
- **HeartDiseaseorAttack**: coronary heart disease [CHD] or myocardial infarction [MI] (0=no, 1=yes)
- **PhysActivity**: physical activity in past 30 days, excluding job (0=no, 1=yes)
- **Fruits**: consume fruit at least once a day (0=no, 1=yes)
- **Veggies**: consume vegetables at least once a day (0=no, 1=yes)
- **HvyAlcoholConsump**: more than 14 or 7 drinks per week respectively for an adult male and an adult female (0=no, 1=yes)
- **GenHlth**: general health level (1=excellent, 2=very good, 3=good, 4=fair, 5=poor)
- **MentHlth**: days of poor mental health in a month (scale 1-30 days)
- **PhysHlth**: physical illness or injury days in past 30 days (scale 1-30)
- **DiffWalk**: difficulty in walking or climbing stairs (0=no, 1=yes)
- **Hypertension**: 0=no hypertension, 1=hypertension
- **Stroke**: (0=no, 1=yes)
- **Diabetes**: 0=no diabetes, 1=diabetes

The dataset contains information regarding 50 thousand patients.

3 Preprocessing

The given dataset was already clean and there were no missing values. We converted the diabetes attribute from a numerical to categorical variable since the models we had taken into consideration required us to do so.

3.1 Model selection

Our goal is to choose the model which yields the lowest possible logloss score. To do so, we took the following steps:

- we chose the models to test. In particular:
  - decision tree
  - simple logistic
  - logistic
  - support vector machine
  - multi layer perceptron
  - naive Bayes
  - bayesian network
- for each model we selected the attributes to include according to the

wrapper approach

- we trained and tested all the models, by implementing a cross validation procedure
- for each model we calculated the logloss score

3.2 Feature selection

An important step to take is understanding what attributes are relevant in the diabetes prediction. We opted to use the wrapper approach, meaning that the classifiers themselves are used to evaluate which features are the most relevant. For each model we obtained a different subset with the relevant parameters. Then we calculated the logloss score in two different situations: when using the selected parameters and when using all of them. We obtained the best results when using the selected parameters, so we added a column filter to discard the irrelevant parameters from the dataset

3.3 Training model and comparison

In order for the performance scores to be influenced as little as possible by the partitioning choice, we trained and tested the models by implementing a cross validation procedure: for each model, we performed 10 validations. Once we obtained the predictions and their related probabilities, we used a “logloss score” as a metric to evaluate the performance of the models. The “logloss ” is calculated by using the following formula[1]:

$$-y_i \cdot log(p(y_i)) - (1 - y_i) \cdot log(1 - p(y_i))$$

The logloss formula isn’t designed to calculate limit cases, where the predicted probability of a label is either 1 or 0. Particularly, in cases where the predicted probability is zero, the logloss cannot be calculated, since $log(0)$ yields minus infinity. However, the limit cases are present in our dataset. To solve this issue, we added the value $1 \cdot 10^{-15}$ to the formula in order to be sure the argument of the logarithm is always strictly greater than zero. In order to obtain the overall logloss score yielded by the model, it is first necessary to calculate it for every record, and then to compute the average. The aim is to achieve the lowest possible logloss score: the lower the logloss score, the higher the probability of getting a correct prediction. The logloss values obtained for the models we considered are reported in table 1.

Table 1: logloss values

model	**logloss score**
Decision tree	8
Simple logistic	0.52
Logistic	0.52
Support vector machine	0.78
Multi-layer perceptron	0.53
Naive Bayes	0.72
Bayesian network	0.51

3.4 Output of the procedure

In the end we compared the logloss scores of all the models, and found out that the one with the lowest score is the Bayesian Network. Another indicator we checked was the confusion matrix and the related accuracy value. We obtained 75%, which we deemed satisfactory.

[1] y_i indicates the actual class value whereas $p(y_i)$ corresponds to the prediction probability of the given class value.

4 The model

The Bayesian Network model is a generalized version of the Naive Bayes model. Naive Bayes is a probabilistic machine learning model that is based on Bayes Theorem. This theorem allows to compute the conditional probability and states that:

$$p(A|B) = \frac{p(B|A) \cdot p(A)}{p(B)}$$

[2] A Naive Bayes model in our case computes the probability of suffering or not from diabetes through the Bayes theorem. The option with the highest probability is chosen as the prediction for the corresponding record. The Naive Bayes model assumes that all features of a data point are independent from one another, while a Bayesian Network allows dependencies between the variables.

We measured the performances of the Bayesian Network model by testing it with a cross validation procedure from the training dataset. We considered the following evaluation parameters:

- *Logloss score*: 0.51
- Accuracy: 74.6%
- ROC curve

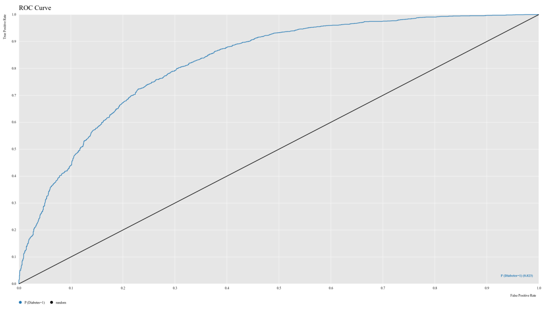

Figure 1: ROC curve

The ROC curve is shown in figure 1. The ROC curve is a plot of the *sensitivity* (the true positive rate) against the *specificity* (the false positive rate). Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. *Sensitivity* and *specificity* represent the probabilities that the model predicts respectively true positives and false negatives. The more the ROC curve hugs the top left corner of the plot, the better the model classifies the data. The ROC curve in figure 1 tells us that the model works well and in order to quantify this perception we must compute the area under the curve (AUC): the closer the AUC is to 1, the better the model. The AUC that we obtained is 0.823, which we consider a satisfactory result.

5 KNIME workflows

5.1 The training workflow

The training workflow is used to learn the model on the training dataset. The workflow is composed of an excel reader node which receives in input the training dataset, subsequently the diabetes attribute is converted from numerical to string format. Then, we applied the bayes network classification algorithm to the whole training dataset after having removed the irrelevant attributes previously identified in the feature selection procedure. The model is then saved through the model writer node on the Knime Hub Public repository. This node cannot be executed by the user since it saves the model online on our team member's account, which cannot be accessed without private credentials. If the model were to

[2] A and B indicate two different events.

be trained with different parameters, it's possible to change the file destination by changing the configuration of the model writer node, and to save the model in a desired location.

5.2 The deployment workflow

This workflow takes as input the trained model, saved in the KNIME Hub Public repository, as well as the test dataset, which is structured in the same way as the training dataset, except for the fact that the target variable is missing because it needs to be predicted. A weka predictor node receives the input sources, and computes the predicted value of diabetes for each record. The result is a dataset containing the explanatory variables, the probabilities of having and not having diabetes, and the final prediction. The workflow produces three results:

- **logloss score**: we compute the logloss value for each record, and then obtain the logloss score by computing the mean of all the logloss values
- **predictions table**: we filter attributes in order to obtain a table containing only the patient ID and the predictions;
- **confusion matrix**: through a scorer node we can compute the confusion matrix, to be sure that the model works fine and produces a reasonable number of accurate predictions.

Please note that the model reader node reads the model from the KNIME account of our team member: if the user changes the location for saving the classification model, the location from which the model is read must be changed accordingly.

6 Data App

After building the training workflow and the deployment workflow we worked on the design of a KNIME data app. A data app in KNIME refers to a standalone application built using the KNIME Analytics Platform. It allows users to interact with and analyze data through a user interface. For the data app to function, it was necessary to build a specific KNIME workflow that grouped all the views into appropriate component nodes. Each component is associated with a web page and can be integrated with widgets that allow user interaction. The data app we designed allows the user to choose whether to upload a file (i.e. a test dataset) or proceed using the training file that was provided to us. The file can be uploaded via a designated icon. The file must have an *.xlsx* format as well as the same structure as the training dataset we worked on. If the user doesn't upload any file, the training dataset is used as default option. The user can select among the three options in a drop-down menu and, once specific content has been viewed, it is possible to go back by pressing the "back" button at the bottom left of the page, and select another option. The options are the following:

- The first option "Database overview" allows the user to perform exploratory visual analysis on the uploaded dataset.
- The second option "Correlations" allows the user to compare, for each variable, its distribution among peo-

ple with and without diabetes. This provides a first idea of the correlation between the dataset variables and the target variable (i.e. diabetes).

- The third and final option “Prediction model & evaluation” allows the user to run the model on the selected file and to view the classification results. Different evalutation methods were considered: the logloss score, the accuracy and the AUC parameter (coming from the ROC curve).

After building the workflow for the Data App and properly configuring the visualization of the component nodes, we uploaded the Data App to a KNIME Server. The Data App can be viewed at this link.

7 Conclusion

We have developed a machine learning classification model for diabetes using the KNIME platform. Our project was structured in three phases: a training workflow where a model was selected and trained on a training dataset, a deployment workflow where the trained model can be used to make predictions on a test dataset, and a data app interface that provides a visual representation of the model's output.

Overall, our project demonstrates the value of using machine learning to classify diabetes based on a set of attributes. Furthermore, we were able to evaluate the performance of the model with the logloss score and determine its effectiveness at predicting diabetes.

Additionally, the use of a data app interface makes the model's output accessible to a wider audience, beyond those with technical expertise in data science. This can be especially valuable in a healthcare setting, where clinicians and patients may benefit from a visual representation of the model's predictions.

In conclusion, our project showcases the application of machine learning in the field of healthcare, specifically in the classification of diabetes.