Veridical Data Science Course Project

Risk of diabetes Predicting Depends on National Health and Nutrition Examination Survey Data

Xinyang Xiong, *ShanghaiTech University*

Abstract—This project focuses on developing a binary prediction algorithm to assess the risk of diabetes in individuals using data from the National Health and Nutrition Examination Survey (NHANES) 2016 dataset. The NHANES, which combines interviews and physical examinations, provides a comprehensive view of the health and nutritional status of both adults and children in the United States. The primary objective is to leverage machine learning techniques—Logistic Regression (LR), Random Forest (RF), and Gradient Boosting Decision Trees (GBDT)—to predict diabetes risk. The project includes several stages: data cleaning and preprocessing, exploratory data analysis, and predictive modeling. In the predictive analysis phase, the aforementioned algorithms are trained and evaluated within a Performance Comparison Score (PCS) framework to assess their effectiveness. The model development and evaluation are carried out using Python 3.8 on an Ubuntu 24.04 system, and the source code is made publicly available on GitHub. This project aims to provide a reliable, data-driven tool to help identify individuals at risk for diabetes, with potential applications in public health and preventative care strategies.

The project aims to generate a binary prediction algorithm that can predict when a patient is at risk of diabetes. The data used is colleted from the 2016 release of the National Health and Nutrition Examination Survey (NHANES), a program of studies designed to assess the health and nutritional status of adults and children in the United States.

The predicting algorithms includes Logistical Regression (LR), Random Forest (RF) and Gradient Boosting Decision Tree (GBDT).

This project contains: Data Cleaning and Preprocessing, Exploratory Data Analysis, Predictive analysis and PCS evaluation. In the predictive analysis part, all the algorithms mentioned above are used, their performance are compared under PCS framework.

All the code are implemented by python 3.8 in ubuntu 24.04 environment, the code is available in https://github.com/LastQuater/vds-final-project

Dataset Description

The National Health and Nutrition Examination Survey (NHANES) 2013-2014 is a comprehensive program designed to assess the health and nutritional status of both adults and children in the United States. This survey is distinctive in its combination of interviews and physical examinations to gather data. Managed by the National Center for Health Statistics (NCHS), part of the Centers for Disease Control and Prevention (CDC), NHANES has been a crucial tool for producing vital health statistics since its inception in the early 1960s.

The survey includes a diverse range of health topics, adapting over time to address emerging health concerns. In 1999, NHANES transitioned to a continuous program that annually examines a representative sample of about 5,000 individuals across the nation. This sample is drawn from 15 different counties each year, ensuring broad geographic coverage.

The National Health and Nutrition Examination Survey (NHANES) 2013-2014 data was collected through a combination of interviews and physical examinations. A representative sample of about 5,000 people from 15 U.S. counties was surveyed each year. Interviews

gathered information on demographics, health behaviors, and dietary habits, while physical exams included measurements like height, weight, blood pressure, and lab tests on biological samples (blood, urine). The data is used to assess the health and nutritional status of the U.S. population.

Data Clearning & Pre-processing

The raw data has 33028 data points and 805 features. In this project, only 11 features are used to avoid great workload. The main file of this part is *dslc_documentation/01_cleaning.ipynb*.

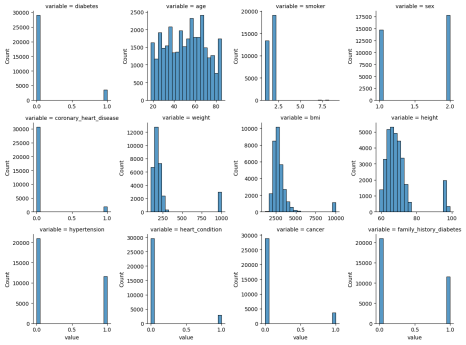

FIGURE 1. Origina data distribution

The original data set distribution is shown in figure1. There are totally 7 category variables and 4 numerical variables. Firstly split the data into training data and validation data. Notice that the label(diabetes) is imbalancing, when splitting, we keep the distribution of label both in training and validation data. Then clean the numerical variables. There are no nan values in the dataset, so the main task is to deal with these invalid values.

The document of the dataset describes meaning of each variable along with the meaning of each value. In the variables selected, values with the meaning such as "Do not know" or "Refused" are treated as invalid values, these values are dropped from dataset in default. Judgement call are used here, and further discuss would be done in the stability analysis. More details are in the code files.

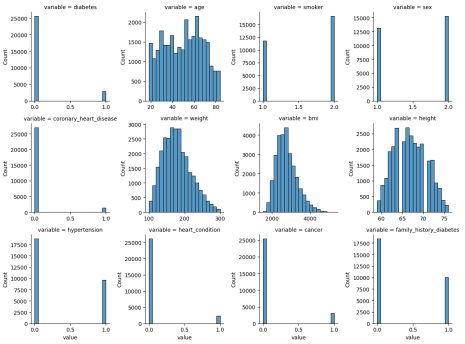

FIGURE 2. Cleaned Data distribution

The cleaned data distribution is shown in figure2, which has 28451 samples and 11 features.

Category variables are encoding into one-hot vectors since logistic regression are used, the final preprocessed dataset has 28451 samples and 20 features.

Exploratory Data Analysis

In EDA, relation between variables are mainly discussed. The file of this stage is *dslc_documentation/02_eda.ipynb*. Figure 3 shows correlation matrix of all numerical variables. Notice that there exists strong relationship between bmi, weight and height. In fact, bmi is an index calculated by weight and height. Relation between variables may influence model's predictive ability. So in training stage, either bmi or height-weight pairs are keep to train. This is also done by a judgement call.

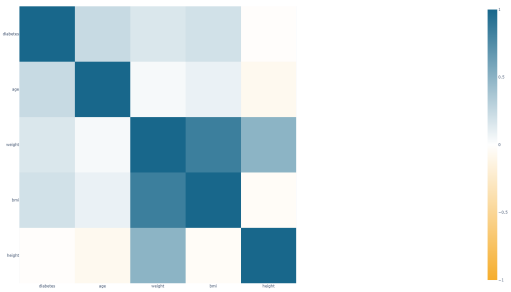

FIGURE 3. Correlations between variables

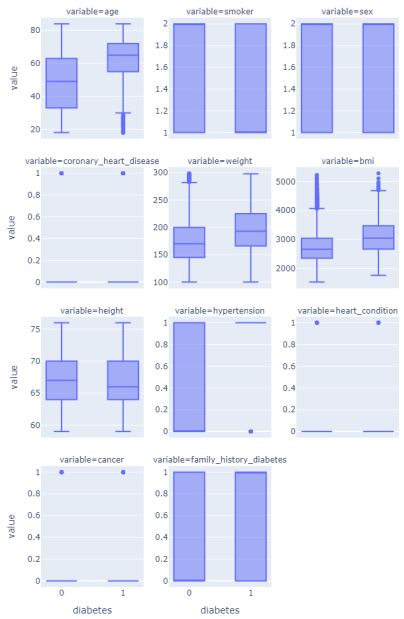

FIGURE 4. Relation between labels and features

Figure 4 shows distribution between labels and features. Notice many features are in balance given different labels, which means labels may not have much with these variables. So in training stage, L1 regularization is taken into consider to "select variables".

Model

The problem is to predict a binary response variable "diabetes". Methods includes: Logistic Regression (LR), Random Forest (RF) and Gradient Boosting Decision Tree (GBDT).

Model Introduction

LR is a statistical method used for binary classification tasks, where the goal is to predict one of two possible outcomes. It models the relationship between a dependent variable (binary outcome) and one or more independent variables (predictors) by estimating probabilities using a logistic function. The core idea is to predict the probability that an instance belongs to a particular class. The model transforms linear combinations of the predictors through the logistic function, which outputs values between 0 and 1, making them interpretable as probabilities. Mathematically, the logistic regression function is:

$$P(y = 1 \mid X) = \frac{1}{1 + e^{-(b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n)}}$$

where y is the binary outcome, $X_1, X_2, \ldots, X_n$ are the predictor variables, and $b_0, b_1, \ldots, b_n$ are coefficients. Logistic regression is widely used in fields like healthcare, marketing, and finance for classification problems.

RF is an ensemble learning method used for both classification and regression tasks. It builds multiple decision trees during training and merges their predictions to improve accuracy and reduce overfitting. The key concept behind Random Forest is the idea of "bagging" (bootstrap aggregating), where random subsets of the data are sampled with replacement to train each tree, and each tree is built using a random subset of features. This randomness helps in creating diverse trees, leading to a more robust and generalizable model.

Each decision tree in the forest makes a prediction, and the final output is determined by aggregating the predictions—either by voting (for classification) or averaging (for regression). Random Forests are known for their high accuracy, ability to handle large datasets with many features, and their resilience to overfitting, making them a powerful tool in machine learning applications like finance, healthcare, and image processing

GBDT is an ensemble machine learning technique used primarily for classification and regression tasks. It builds a series of decision trees sequentially, where each new tree corrects the errors of the previous ones. The key idea behind GBDT is gradient boosting, where the model is trained to minimize a loss function by fitting the residuals (errors) of the previous trees.

At each step, a new tree is trained to predict the gradient (negative residual) of the loss function with respect to the model's predictions. This process continues iteratively, adding trees that improve the model's accuracy. The final prediction is the weighted sum of all the trees' outputs.

GBDT is highly effective at handling complex, nonlinear relationships and works well with both small and large datasets. It is particularly popular for its accuracy and versatility in real-world tasks, such as in Kaggle competitions, financial forecasting, and recommendation systems.

All the model used are in python module it scikit-learn.

Model Selection

Each model mentioned above is trained with training data and tested with validation data. And cross validation is used to select hyperparameters. Grid search is applied for LR since its parameter space is small, while random search is applied for RF and GBDT due

to their large parameter space. The hyperparameters grid are shown in table 1, 2, 3

After this, the optimal parameters are used to show the results.

Results

Below will show the model's performance on validation dataset.

Metrics

the metrics include accuracy score, f1 score and roc curve. Accuracy score is the rate of true predict number:

$$Acc = \frac{\#\ \text{true predict}}{\#\ \text{sample}}$$

F1 score is the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

ROC curve is a graphical representation of a classifier's performance across different classification thresholds, showing the trade-off between true positive rate (TPR) and false positive rate (FPR) at various threshold settings. And the area under curve (AUC) indicates models' perfomance, which will be also displayed.

Thresholds Select

Since thresholds influence accuracy and f1 score model, a distribution density graph is used to select an "optimal" threshold. In this project, the cross point in the graph is considered to be an "optimal" threshold. All the results showing below follow this threshold. The distribution density graph is shown in figure 5, 6, 7

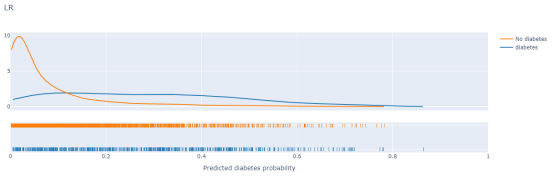

FIGURE 5. Distribution density plot of logistic regression

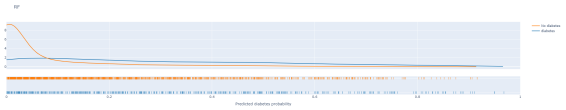

FIGURE 6. Distribution density plot of random forest

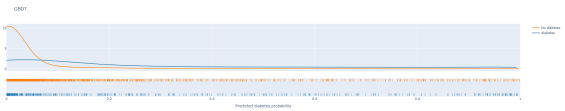

FIGURE 7. Distribution density plot of gradient boosting decision tree

Results display

Roc curves of the three models are shown in figure 8. In this metric, LR performs best because the area below the curve is the largest. RF and GBDT perform similarly.

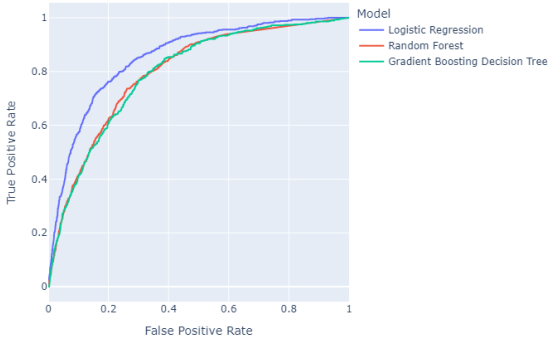

FIGURE 8. Roc curves of models

Table 4 shows accuracy and f1 score under the "optimal" threshold. LR has the best f1 score while GBDT has the best accuracy score. AUC of LR are the best, RF and GBDT have similar perfomance.

Model	Acc	F1	AUC
LR	0.76	**0.40**	**0.854**
RF	0.73	0.35	0.797
GBDT	**0.80**	0.36	0.793

TABLE 4. Accuracy , f1 scores and AUC of models

Discussion

Stability on perturbed data

Create 100 perturbed versions of training data, and training 100 models to evaluate the stability, the ROC curves are shown in figure 9, 10, 11.

Hyperparameters	Meaning	Value
C	Inverse of regularize strength	[0.01, 0.1, 1, 10, 100]
solver	Algorithm to use in the optimization problem	['lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag', 'saga']
penalty	Regularize method	['l1', 'l2']

TABLE 1. LR hyperparameters grid

Hyperparameters	Meaning	Value
n_estimators	Number of trees	Integer between [10, 200]
max_depth	Max depth of trees	[None, 5, 10, 15, 20, 30, 40]
min_samples_split	The minimum number of samples required to split an internal node	Integer between [2, 20]
min_samples_leaf	The minimum number of samples required to be at a leaf node	Integer between [1, 20]
max_features	The number of features to consider when looking for the best split	['auto', 'sqrt', 'log2']
bootstrap	Whether bootstrap samples are used when building trees	[True, False]

TABLE 2. RF hyperparameters grid

Hyperparameters	Meaning	Value
n_estimators	Number of trees	Integer between [50, 200]
max_depth	Max depth of trees	[3, 5, 7, 10]
learning_rate	Learning rate in each tree	U(0.01, 0.2)
sub_sample	The fraction of samples to be used for fitting the individual base learners	U(0.3, 0.7)
min_samples_split	The minimum number of samples required to split an internal node	Integer between [2, 20]
min_samples_leaf	The minimum number of samples required to be at a leaf node	Integer between [1, 20]

TABLE 3. GBDT hyperparameters grid

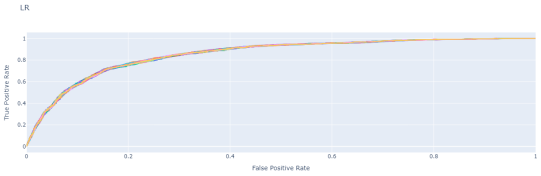

FIGURE 9. Roc curves on perturbed data with LR

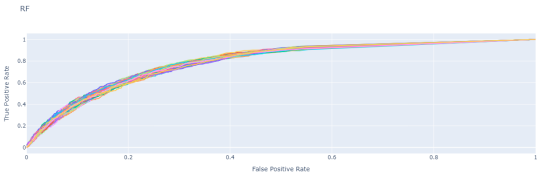

FIGURE 10. Roc curves on perturbed data with RF

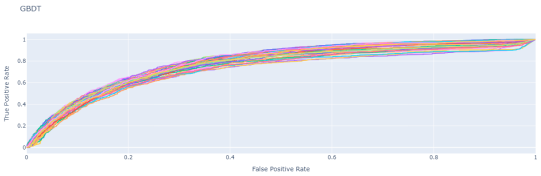

FIGURE 11. Roc curves on perturbed data with GBDT

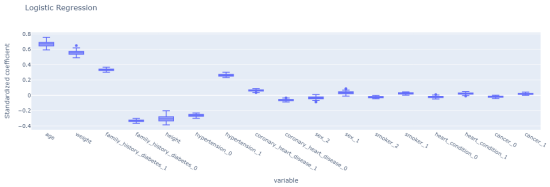

FIGURE 12. Standardized coeeficient of variables

The ROC curves of LR looks most tight, which means it has best stability to data, while RF has the second stability, GBDT has the least stability. This occurs may because LR takes complete grid search, while RF and GBDT takes random grid search, not always best parameters are set. More over, ensemble learners have much more varibles than LR, which will definitely increase the variance of model.

True positive rate, true negative rate and accuracy on perturbed data are also compared, showing in figure 13, 14, 15. In these figures, GBDT has worst stability while LR and RF perform better. Specifically, GBDT has low true positive rate but high true negative rate and accuracy score. But positive data in labels is 9 times to negative labels in our data, so GBDT may learn some deep relations between the features and labels. In general, LR still performs best in this dataset.

We also use standardized perturbed data to evaluate the importance of each variables, result is shown in

12. The result follows our analysis in EDA: much variables have not much to the labels, only age, weights, family_history_diabetes, height and hypertension have non-negligible coefficient.

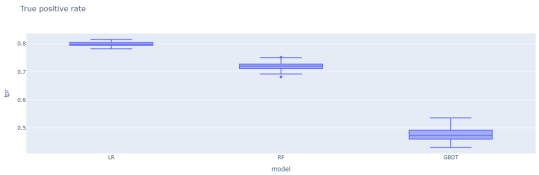

FIGURE 13. True positive rate on perturbed data

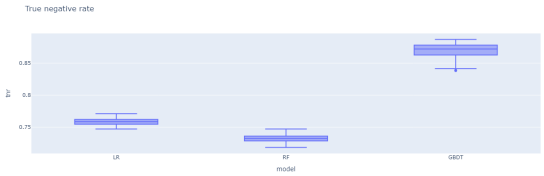

FIGURE 14. True negative rate on perturbed data

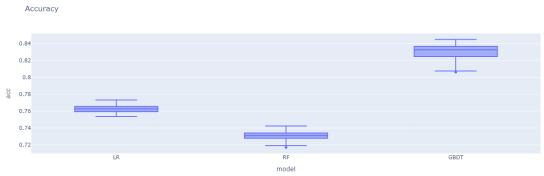

FIGURE 15. Accuracy on perturbed data

Stability on judgement call

Create different versions of training data according to all judgement call mentioned above, the ROC curves are shown in figure 16, 17, 18. In this stage, LR seems not as stable as RF and GBDT. This implies LR is more sensitive to varibles selection since our judgement call is mainly used to select features.

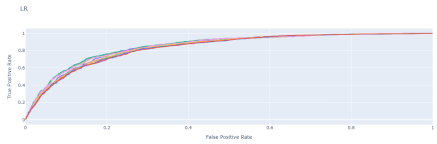

FIGURE 16. Roc curves on judgement call with LR

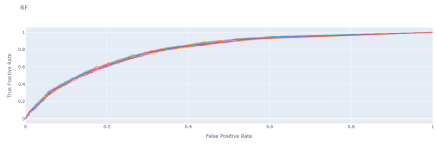

FIGURE 17. Roc curves on judgement call with RF

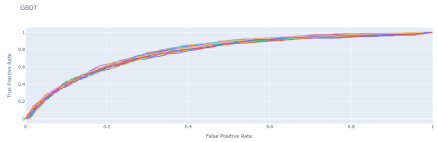

FIGURE 18. Roc curves on judgement call with GBDT

Similar results also seen in the true positive rate, true negative rate and accuracy figures 19, 20, 21. Other conclusions remains the same to perturbed data.

FIGURE 19. True positive rate on judgement call

FIGURE 20. True negative rate on judgement call

FIGURE 21. Accuracy on judgement call