Cardiovascular Risk Identification Using Machine Learning Methods

Pian Wan* Qianjun Xu* Siyuan Cheng*
{pian.wan, qianjun.xu, siyuan.cheng}@epfl.ch
EPFL, Switzerland

***Abstract*—Machine learning has attracted a lot of attention in data analysis as more and more data are becoming available. In this essay, we apply machine learning methods to analyze the 2015 Behavioral Risk Factor Surveillance System (BRFSS) dataset and try to predict the cardiovascular risks for new data. We demonstrate the effectiveness of data pre-processing, model selection, and hyper-parameter tuning in our method by comparing. Our method finally achieved an F1 score of .432 and an accuracy of .883 on the test set by AIcrowd. Code and dataset are available: https://github.com/epfml/ml-project-1-cross-entropy/tree/main.**

I. Introduction

Cardiovascular disease remains a significant cause of mortality among humans. Identifying high-risk individuals is crucial to prevent the onset of heart disease. The dataset originates from the 2015 BRFSS survey, which involves over 400,000 respondents from the United States. The goal of this work is to use Machine Learning models to predict whether an individual suffers from cardiovascular disease from their provided data at an early stage. To achieve this, we evaluate different regression models and employ several data pre-processing techniques, aiming to harness the full potential of this dataset.

II. Data Pre-processing

In this section, we introduce some data pre-processing techniques to reduce the potential interference from irregularities in the raw data. Our pipeline follows the order A-B-CD-E. Fig. 1 shows some raw and processed data.

A. Data selection

To take full advantage of the information contained in the dataset, we decide to start with all the variables provided, while eliminating the variables that are only related to the questionnaire itself (e.g. _STATE, FMONTH), with over 60% missing values (e.g. INSULIN, FEETCHK2), and too complicated to be one-hotted (e.g. EXRACT11). After the selection, there are a total of 261 features in the final scope of variables. Also, all the 328135 samples in the dataset are preserved to maximize the training data.

B. NaN and special values

Given a feature vector, we mark it deleted if there are more than N NaN values in the vector. The deleted features will not be processed further in our pipeline. For other NaN values, we use the mean or carefully selected value as its

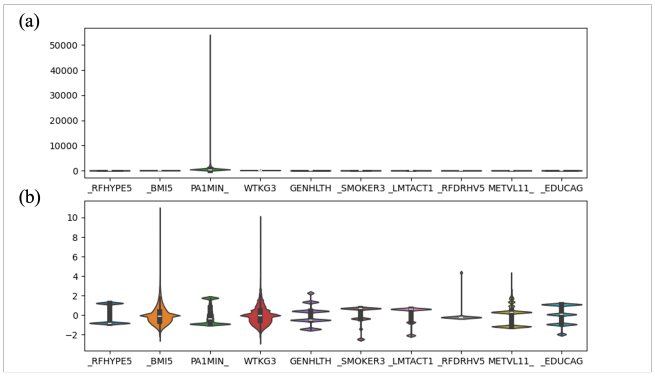

Figure 1: **The distribution of some features before/after pre-processing.** (a) is raw, and (b) is the processed data.

value. Some input features (e.g. _RFHLTH, _RFHYPE5, WTKG3) use special values such as 9 or 9999 to represent error values (i.e. Refused, Not Sure, Missing). We recognize these values the same as NaN values if they cannot be treated to our one-hot encoder. In our pipeline, we choose $N = 10$.

C. Normalization and standardization

The values of features may have different scales and units. To make sure the features contribute evenly to the model and speed up the convergence speed of our optimization algorithms, we apply normalization or standardization to our input features. The normalization makes features scaled within a range of $[0, 1]$. The standardization centers the data around zero with a standard deviation of 1, which preserves the shape of the original distribution and provides robustness to outliers. In particular, we compare z-score standardization and max-min normalization to the data.

D. One-hot encoding

Some features use arbitrary integers to represent different states (e.g. for _RACE, "White" is 1, "Black" is 2, "American Indian" is 3...) which may cause the ML model to mistakenly assume an ordinal relationship between the categories. We use one-hot encoding for categorical variables to create binary columns for each category, treating them as distinct categories and ensuring no information is lost after data pre-processing.

E. Data resampling

The label in the dataset is not uniform, there are 28975 positive points and 299160 negative points (about 10x positive points), leading to potential imbalance problems for

training. Handling class imbalance in the training process is essential to ensure that machine learning models are not biased toward the majority class. In the training process, we use oversampling to increase the number of samples in the minority class to 1.8 times that of the majority class.

III. Method and Experiments

In this section, we validate the effectiveness of our proposed pre-processing methods, hyperparameters, and regression models by conducting ablation studies.

A. Experiment metrics

We use F1 score and accuracy as our metrics. Because of the imbalance in the dataset, we prioritize the F1 score over accuracy due to its sensitivity to the minority class.

B. Ablation studies

1) Data pre-processing methods: We compare different pre-processing methods step by step. The logistic regression experiments here apply hyperparameters of $\lambda\_ = 10^{-15}$, max_iteration = 150500, and batch_size = 3234. The cosine annealing scheduler is applied in these experiments with $\gamma_{max} = 10^{-2}$, $\gamma_{min} = 10^{-6}$, $T_{max} = 1000$.

Table I shows that F-RV-H-S has superior performance against others, which is then selected as our final data pre-processing method.

Method	Train		Validation	
	F1 Score	Acc	F1 Score	Acc
F	.214	**.900**	.212	**.898**
F-RV	.288	.613	.290	.611
F-RV-RS	.260	.603	.330	.729
F-RV-H	.288	.612	.290	.610
F-RV-H-N	.417	.849	.416	.853
F-RV-H-S	**.421**	.858	**.422**	.857

Table I: **Ablation studies on data pre-processing methods.** F is the baseline replacing NaN with average, RV replaces missing values with carefully selected values, RS removes sample points with more than 10 missing values, H uses one-hot encoding for categorical discrete variables, N means normalization, and S refers to standardization.

2) Model selection: We summarize the performances of various methods in Table II. Due to a large amount of training data (> 300000), performing full gradient descent is time-consuming, ineffective, and impractical for real-world problems. Also, full gradient descent cannot make good use of the imbalanced data. Among all the methods, logistic regression SGD shows the best performance.

3) Hyperparameter tuning: The given dataset is split into five folds with the k_fold cross-validation method (i.e. $k = 5$). We compute the mean metrics as results by using one of the folds for validation and the rest for training. The computed metrics and then be used to identify the optimal learning rate γ and the regularization coefficient $\lambda\_$. From TableIII, we finally regard the Cosine annealing scheduler[1] with $\gamma = 1 \times 10^{-2}$ and $\lambda\_ = 1 \times 10^{-15}$ as optimal.

Model	Train		Validation		Test	
	F1 Score	Acc	F1 Score	Acc	F1 Score	Acc
linear regression GD	.382	.866	.375	.861	×	×
linear regression SGD	.415	.838	.410	.835	×	×
ridge regression	.366	.865	.368	.865	×	×
least square regression	.367	.883	.364	.881	×	×
logistic regression GD	.355	**.907**	.344	**.904**	×	×
logistic regression SGD	**.425**	.868	**.423**	.866	<u>.432</u>	<u>.883</u>

Table II: **Ablation studies on model selections.** We use the best result that we can derive from the pre-processed dataset using these models. The dataset is split into 8:2 as training and validation set here. We use full train data for AIcrowd testing (i.e. underline).

Hyper parameters			Train		Validation	
Scheduler	γ	$\lambda\_$	F1 Score	Acc	F1 Score	Acc
Fixed	1×10^{-3}	1×10^{-15}	.381	.818	.374	.819
Fixed	5×10^{-3}	1×10^{-15}	.417	.842	.413	.843
Fixed	1×10^{-2}	1×10^{-15}	.374	.761	.375	.761
Linear^1	1×10^{-2}	1×10^{-15}	.416	.838	.411	.838
Linear^2	1×10^{-2}	1×10^{-15}	.421	**.886**	.407	**.887**
Cosine^1[1]	2×10^{-2}	1×10^{-15}	.418	.837	.415	.838
Cosine^1[1]	1×10^{-2}	1×10^{-15}	.416	.839	.412	.840
Cosine^2[1]	2×10^{-2}	1×10^{-15}	.416	.838	.420	.840
Cosine^2[1]	1×10^{-2}	1×10^{-15}	**.424**	.853	**.423**	.849

Table III: **Ablation studies on hyperparameter tuning.** We conduct experiments on fixed and scheduled learning rates. For linear schedulers, we decrease 5×10^{-8} and 1×10^{-8} for 1 and 2 respectively after every step. For cosine schedulers, we choose 1×10^{-5} and 1×10^{-6} as the final learning rate for 1 and 2 respectively with the same T_{max} as III-B1.

C. Results

From Table I, Table II, Table III, we select F-RV-H-S, logistic regression SGD, a cosine learning rate scheduler with $\gamma = 1 \times 10^{-2}$, $\lambda\_ = 1 \times 10^{-15}$, $T_{max} = 1000$ as our final proposed method for submission. The predicted result on the test set was uploaded to AIcrowd, achieving an F1 score of .432 and an accuracy of .883. Submission ID: #243770.

IV. Conclusion

In this essay, we propose an effective method to identify cardiovascular risk from given features. We apply various data pre-processing techniques, including one-hot encoding, null value replacements, normalization, standardization, and resampling. We also study six regression models and try different hyperparameters for training. The ablation studies show the comparison of data pre-processing methods, machine learning models, and hyperparameters. Our final approach combines data pre-processing, logistic regression SGD, and the cosine learning rate scheduler, which achieves the best result with an F1 score of .432 and an accuracy of .883 on the test set by AIcrowd.

References

[1] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.