CS-433 Machine Learning: Project 1 Report

Ilieva Nadezhda, Krsteski Stefan, Tashkovska Matea
Department of Computer Science, EPFL, Switzerland

***Abstract*—This study aims to leverage machine learning techniques, specifically seven distinct models, including a neural network, to predict the risk of Myocardial Ischemic Coronary Heart Disease using the data from Behavioral Risk Factor Surveillance System. With a comprehensive dataset encompassing 328,135 data points and 321 features, our rigorous data processing and analysis led to the best prediction accuracy using a neural network, achieving an F1 score of 0.417 ± 0.004. These findings underscore the potential of utilizing advanced machine learning tools in healthcare to facilitate early disease detection and improved patient care.**

I. Introduction

As Cardiovascular Diseases (CVD) rise globally, the rise of new technologies such as machine learning algorithms can help with their early detection and prevention. The data from Behavioral Risk Factor Surveillance System (BRFSS) includes records of more than 300,000 individuals who were classified as having coronary heart disease (MICHD) if they reported having been told by a provider they had MICHD or if they reported having been told they had a heart attack (i.e., myocardial infarction) or angina. Using this dataset, our main goal is to build a model that can predict the risk of getting MICHD in a particular clinical and lifestyle scenario.

II. Models and Methods

A. Exploratory Data Analysis

In the initial phase of our project, we conducted exploratory data analysis (EDA) to better understand our dataset. The dataset consists of 328,135 data points with 321 features. We counted unique values for each feature, calculated missing value percentages, examined feature correlations, and explored feature distributions. Importantly, we decided not to set fixed thresholds for handling missing values, correlation, or feature type (categorical/continuous) but instead to treat them as hyperparameters, giving us the flexibility to adapt our data preprocessing as we experiment.

B. Feature Processing

In our feature processing phase, we took several systematic actions to refine our dataset. Initially, we dropped columns with missing values surpassing a specified threshold. Subsequently, we categorized features into either categorical or continuous, based on a predetermined threshold. Missing values were imputed with the mean for continuous features and the median for categorical ones. We removed single-valued columns and those with correlation scores exceeding a set threshold. To capture non-linear relationships in our dataset, we incorporated polynomial features. We also experimented with applying a logarithmic transformation to expand the dataset. Furthermore, we calculated the ratio of each feature compared to its average, maximum, and minimum, resulting in three new features for each original one. As a final step of our data preprocessing, we applied standardization, which rescales the data values to have a mean of 0 and a standard deviation of 1.

C. Implementation of Machine Learning Models

As required, we implemented six distinct machine learning models: linear regression using gradient descent, linear regression using stochastic gradient descent, least squares regression using normal equations, ridge regression using normal equations, logistic regression using gradient descent and regularized logistic regression using gradient descent.

D. Hyperparameter Tuning

Given that it is a classification problem, we decided to focus on tuning logistic and regularized logistic regression models. We began by conducting a random search, which involved both data preprocessing and model-specific hyperparameters, as it allowed us to quickly explore the extensive range of options. Following this, we attempted further hyperparameter refinement focusing on the best configurations identified during the initial search.

E. Neural Network

To improve the results obtained with the linear methods, we implemented a feed-forward neural network optimized for binary classification. The architecture begins with an input layer of a desired size. This is followed by a variable number of fully connected layers, each employing the Rectified Linear Unit (ReLU) activation function, with initial weights drawn from a Gaussian distribution. The network ends with an output layer, which predicts the class using the softmax function to transform logits into class probabilities.

Training includes minimizing the cross-entropy loss. Cross-entropy loss, also known as log loss, measures the performance of a classification model. Its formula for binary classification is given by:

$$L(y,p) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(p_i) + (1-y_i)\log(1-p_i)\right]$$

where y represents the true labels, p are the predicted probabilities, and N is the total number of instances. In essence, it quantifies the dissimilarity between the predicted probabilities and the true class labels. For binary classification, a lower cross-entropy value indicates a better-performing model, and it penalizes predictions that are confidently wrong more severely than those that are slightly off.

The data preprocessing step for the neural network models included imputing missing data values with median/mean based on a categorical/continuous threshold of 200 and standardizing the data. To address the imbalance in the dataset, we incorporated class weights to penalize the majority class more severely, thus enhancing the classification of the minority class. Given our dataset's substantial size of approximately 300,000 instances, we experimented with different batch sizes. Finally, we experimented using a different number of layers and epochs.

III. Results

A. Logistic and Regularized Logistic Regression

The best results, based on the local 5-fold cross-validation were obtained using logistic regression with the following settings: a drop threshold for missing values set at 0.8, a cutoff of 50 for categorizing features into categorical or continuous and a correlation drop threshold of 0.8. We excluded the polynomial features but included logarithmic and ratio transformations. The number of iterations was 100 and the learning rate was 0.3. This resulted in an F1 score of 0.346 ± 0.001 and an accuracy of 0.776 ± 0.001.

B. Neural Network

In our exploration of neural networks, we implemented various configurations in order to maximize the F1 score.

Iter.	Weights	Num. Epochs	Batch Size	Num. Layers	CV F1
1	[1, 1]	10	256	3	0.241 ± 0.180
2	[1, 4.5]	10	256	3	0.402 ± 0.006
3	[1, 4.5]	5	256	3	0.408 ± 0.007
4	[1, 4.5]	5	1000	3	0.413 ± 0.002
5	[1, 4.5]	5	1000	2	0.417 ± 0.004

Table I: Neural Network Experiments Results

As shown in Table I, our first neural network model performed poorly, predicting mostly the majority class. We attributed this to the imbalance between the classes- the minority class appears only one-tenth as often as the majority class. To address this, we introduce penalty weights of [1, 4.5], penalizing errors in predicting the minority class (class 1) 4.5 times more than errors in the majority class (class 0). This led to a notable improvement. Upon observing that the model overfits after the fifth epoch, as shown in Figure 1, we decided to cut our training at this point, continuing with only five epochs for the following models. Using a batch size of 1000, as opposed to 256, improved the model performance. A bigger batch size leads to less frequent updates which can stabilize the gradient descent process and reduce the noise in updates. We experimented with decreasing the neural layers to achieve model robustness and simplicity, resulting in our best-performing model. The highest F1 score achieved was 0.417 ± 0.004 on the local 5-fold cross-validation and 0.433 on AIcrowd, showcasing a significant improvement over our logistic regression model.

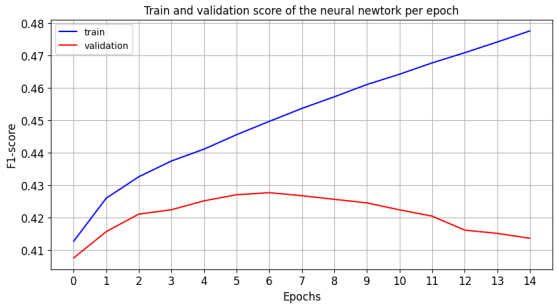

Figure 1: Train and validation F1 score of the neural network per epoch

IV. Discussion

The logistic and regularized logistic regression models, although simpler, showcased the potential of machine learning in predicting MICHD risk. However, the neural network outperformed them, achieving an F1 score of 0.417 ± 0.004, highlighting its potential in handling vast and complex datasets. It's worth noting that while our study focused primarily on linear models and neural networks, other types of models, especially tree-based models like random forest and gradient boosted trees, have been known to perform exceptionally well with tabular data. In conclusion, by effectively predicting the risk of MICHD using the provided dataset, our research contributes meaningfully to the larger goal of early disease detection.

V. Summary

In this report, we used machine learning techniques to assess the risk of getting a heart condition called MICHD using BRFSS data. After preprocessing the data, we tested seven distinct models, including a neural network. The neural network outperformed the machine learning models, with an F1 score of 0.417 ± 0.004.