Optimizers analysis on predicting diabetes using Logistic Regression

Nguyen Quoc Viet, Pham Anh Quan, Pham Trong Dong Hai

Advisor: Do Thai Giang

ABSTRACT

Diabetes mellitus (DM) is a growing public health concern worldwide, contributing greatly to morbidity, mortality, and healthcare loss. Its prevalence is steadily increasing, impacting millions of individuals with serious complications like heart disease, vision loss, lower-limb amputation, and kidney disease. Early detection and intervention are crucial to enhancing patient health and alleviating strain on healthcare resources. In this study, we design a prediction model from scratch based on Logistic Regression, which is a classification model that uses probabilistic estimation, and is extensively used in clinical analysis. We will also integrate different optimizers (Batch, Mini-Batch, Stochasitc Gradient Descent with momentum & decay) and regurlarization techniques into the model and observe how well the model works with each optimizer. The study mainly used the BRFSS2015 dataset from CDC's (Centers for Disease Control and Prevention) clean survey in 2015. We performed EDA and Feature Selection methods, such as VIF (Variance Inflation Factor) and Chi-square test which indicated 16 important factors. Additionally, undersampling was also applied to deal with dataset being unbalanced. The study concluded that our model works generally well with all of the optimizers with accuracy all yield at 73%, however, based on several evaluation metrics, Mini-Batch comes out as the best optimizer for this study. With this result, it is safe to argue that our model could potentially be applied to return a reasonable prediction on whether a person has diabetes or not.

Keywords: Diabetes mellitus, Machine Learning, Feature Engineering, Undersampling, Logistic Regression, Optimizers.

1 Introduction

The World Health Organization (WHO) estimated that Diabetes Mellitus (DM) is the direct cause of approximately 1.6 million deaths in 2021 [1]. Diabetes is a disease when the level of glucose in a person's blood is too high. According to medical experts, type 1 diabetes is typically the condition when the pancreas is unable to produce sufficient insulin, and type 2 diabetes is characterized by the inability of the body to utilize insulin [2]. During digestion, glucose is released into the bloodstream and insulin is responsible for facilitating the uptake of glucose into cells for energy production. However, when insulin production is inadequate, or cellular insulin resistance occurs, glucose remains in the bloodstream, resulting in hyperglycemia [3].

There are various symptoms that can be associated with persistent hyperglycemia, including excessive thirst, frequent urination, and increased hunger. A normal person typically has blood glucose level ranging between 70 and 99 mg/dL, while an elevated glucose level (above 126 mg/dL) is often indicative of diabetes. Individuals with glucose concentrations in the bloodstream lying between 100 and 125 mg/dL are classified as prediabetic [4]. If not properly controlled, diabetes can be the onset of several health problems, namely cardiovascular disease, stroke, nerve damage, and kidney failure [5]. Currently, there is no definitive cure for diabetes [6]. People affected by prolonged diabetes may have a high chance of suffering from macrovascular and microvascular problems. Macrovascular complications consist of damage to large blood vessels, which is likely to result in heart disease, stroke, and peripheral artery disease. On the other hand, microvascular problems are those related to small blood vessels, which directly affect the kidneys, eyes, nerves, and extremities [7]. However, diabetes can be effectively managed provided that early diagnosis is conducted, and can be mitigated with a healthy lifestyle, including regular physical activity and a well-balanced diet [8]. As for pre-diabetic individuals, Type 2 diabetes can be prevented by maintaining a healthy weight and following a physically active routine. There have been several programs such as the National Diabetes Prevention Program, which is initiated by the Centers for Disease Control and Prevention (CDC), promoting lifestyle modifications to reduce the risk of this disease [9]. There is a large amount of data related to the healthcare sector, including patient records, hospital logs and diagnostic results. Traditionally, physicians' clinical expertise is decisive in the diagnosis of diseases. Despite being valuable, their insights are prone to human error and subjectivity, which results in the overlooking of critical patterns and potentially delays appropriate treatment. Therefore, there is a need for automated techniques for accurate and early diabetes detection. Recently, data mining and machine learning are gaining popularity in medical research because of their value and effectiveness. Data mining techniques are crucial for efficient preprocessing and selection of predictive healthcare features, while machine learning models allow the diabetes prediction process to be partially automated [10]. These techniques are valuable in uncovering hidden patterns in medical data, leading to more accurate diagnoses. Several methodologies are integrated with data mining, including machine learning, statistical analysis, and database management in order to extract meaningful insights from large-scale datasets [11]. According to Nvidia, machine learning can help improve diagnostic accuracy by processing structured data, learning underlying trends, and creating predictive models [12].

There are several machine learning approaches that have been proven effective in predicting diabetic individuals. One of the most commonly used methods is logistic regression, which is a fundamental classification algorithm typically dealing with binary classification tasks. In this study, we are motivated to design efficient methods to improve diabetes prediction by analyzing different optimization techniques for logistic regression models. By implementing logistic regression from scratch and integrating different optimizers-stochastic gradient descent (SGD), mini-batch gradient descent, and batch gradient descent-we aim to enhance model accuracy and computational efficiency in handling medical datasets.

Our approach takes different health indicators as input and predicts the likelihood of diabetes as output. By comparing the performance of different optimization techniques, we evaluate their impact on convergence speed, computational cost and predictive ability. This study contributes to the development of efficient machine learning methodologies for diabetes prediction, offering insights into selecting the most suitable optimizers for medical datasets.

2 Related Works

Machine Learning and Deep Learning techniques have been widely used by various researchers in the pursuit of aiding clinical institutions in diagnosing and detecting diabetes. Aishwarya and Vaidehi[13] used various models such as Support Vector Machines, Random Forest Classifier, Decision Tree Classifier, Extra Tree Classifier, Ada Boost algorithm, Perceptron, Linear Discriminant Analysis algorithm, Logistic Regression, K-NN, Gaussian Naïve Bayes, Bagging algorithm and Gradient Boost Classifier. They used two datasets: PIMA Indian Diabetes dataset for training and another Diabetes dataset for testing the trained models. Logistic Regression returned an accuracy of 96% on the test dataset.

Khanam, Jobeda Jamal, and Simon Y. Foo[14] also used different machine learning algorithms: Decision Trees, K-Nearest Neighbor, Random Forest, Naive-Bayes, Logistic Regression, AdaBoost and Neural Network. They performed data preprocessing methods on the dataset. The study shows that Logistic Regression works well on diabetes prediction with 76.82% accuracy on K-fold and 78.85% accuracy on splitting. However Neural Network with 2-hidden layers outperformed Logistic Regression with 88.6% accuracy.

Joshi and Dhaka[15] chose two algorithms-Logistic Regression and Decision Tree to predict diabetes on the PIMA Indian Diabetes dataset. Missing values in predictors have been imputed with the median. For Logistic Regression, they fitted multiple models using all of the possible combinations of predictors to select the best model based on different criteria. Furthermore, interaction terms were also added to monitor their impact on performance. The final Logistic Regression returned an accuracy at 78.26% with a 21.74% cross-validation error rate. The alternative model, Decision Tree, was also constructed, their study experimented different tree complexities(6 nodes, 10 nodes, 13 nodes), which observed that a 6-node tree have the lowest residual deviance. The 6 nodes Decision Tree model yielded a 74.48% accuracy.

Rajendra, P. and Latifi, S.[16] mainly used Logistic Regression on PIMA and Vanderbilt datasets. Feature selection was carried out using two methods. Ensemble methods are further used to enhance better performance compared to a single model. They obtained the highest accuracy of 78% for Dataset 1, where they employed the ensemble technique: Max Voting. For Dataset 2, they performed Max-Voting and Stacking, achieving a 93% accuracy.

On the other hand, MOHAMED, M.H., KHAFAGY, M.H., MOHAMED, N., KAMEL, M. and SAID, W.[17] also used different machine learning models including Decision Tree, Logistic Regression, AdaBoost, Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LGBM), CatBoost Classifier, Support Vector Machine, and Linear Discriminant Analysis (LDA) and a blended model combining AdaBoost and Logistic Regression. However different from the previous studies, they used the Behavioral Risk Factor Surveillance System (BRFSS) dataset from 2014 and 2015. Preprocessing and oversampling (SMOTE) techniques were applied to handle class imbalance. The Logistic Regression achieved a moderate accuracy of 80.68% for 2014BRFSS dataset and 73% accuracy for 2015BRFSS dataset.

The studies above provided a comparative performance evaluation among several machine learning algorithms. Most of them used PIMA Indian Dataset to analyze, however common limitations surfaced around these papers where the sample size of this dataset is relatively small. Data pre-processing and cross-validation techniques are often carried out to enhance accuracy, while feature selection methods are used to not only indicate predictors but as an approach to reduce the dimension of the dataset. In this paper, we concentrated on a single model integrated with optimizers, we also examined data pre-processing, feature selection and undersampling techniques to enhance the final accuracy while improving execution speed.

3 Data Preparation

3.1 Overview of Datasets for Diabetes Prediction

The Behavioral Risk Factor Surveillance System (BRFSS) is one of the largest ongoing health surveys globally, collecting data on health-related risk behaviors, chronic health conditions, and preventive services [18]. Numerous studies on public health risk assessment have leveraged BRFSS due to its extensive sample size and diverse demographic representation [19, 20, 21]. While multiple datasets exist for diabetes prediction, BRFSS provides one of the most comprehensive and widely used datasets. Table 1 presents a comparison of publicly available datasets relevant to diabetes prediction.

Dataset	**Year**	**Sample Size**	**Features**	**Data Type**	**Accessibility**
BRFSS2015 (Cleaned)	2015	253,680	22	Survey responses	Public (Kaggle)
NHANES	2015	$\sim$10,000	100+	Clinical & survey	Public (CDC)
Pima Indian Diabetes	1990	768	8	Clinical	Public (UCI)

Table 1: Comparison of Publicly Available Datasets for Diabetes Prediction

3.2 BRFSS2015 Dataset Composition

The dataset used in this study is the cleaned version of the BRFSS2015[22] dataset available on Kaggle. The cleaned dataset consists of 253,680 survey responses, reduced from the original 400,000+ due to preprocessing and removal of incomplete records. The dataset includes key health-related attributes such as Body Mass Index (BMI), physical activity levels, smoking habits, alcohol consumption, and diabetes status [23]. The dataset contains a total of 22 features, including the target variable, which is primarily used for diabetes prediction. The target variable includes a classification label, such as a diabetes diagnosis, making it suitable for predictive modeling.

Since BRFSS is a publicly available dataset curated by the CDC, no additional data collection was conducted for this study.

3.3 Dataset Illustration

The dataset consists of structured tabular data. Unlike image or text-based datasets, health survey datasets like BRFSS rely on structured numerical and categorical responses. Table 2 illustrates a subset of the dataset highlighting key attributes.

Diabetes_binary	HighBP	HighChol	CholCheck	BMI	Smoker
0.0	1.0	1.0	1.0	40.0	1.0
0.0	0.0	0.0	0.0	25.0	1.0
0.0	1.0	1.0	1.0	28.0	0.0
0.0	1.0	0.0	1.0	27.0	0.0
0.0	1.0	1.0	1.0	24.0	0.0
0.0	1.0	1.0	1.0	25.0	1.0
0.0	1.0	1.0	1.0	30.0	1.0
0.0	1.0	1.0	1.0	25.0	1.0
1.0	1.0	1.0	1.0	30.0	1.0
0.0	0.0	0.0	1.0	24.0	0.0

Table 2: Sample rows from BRFSS2015 showing selected health indicators

3.4 Data Preprocessing and Feature Engineering

3.4.1 Missing values and outliers identification

As the aforementioned BRFSS2015 survey has been well-cleaned and structured into tabular format, it reduces the need for dealing with missing values. We have identified 24,206 duplicated records in the dataset and removed them to prevent bias and data leakage when building our models. Since most of the predictors are binary values, boxplots were chosen to detect outliers for the rest of the numerical features and the result is visualized in Figure 1.

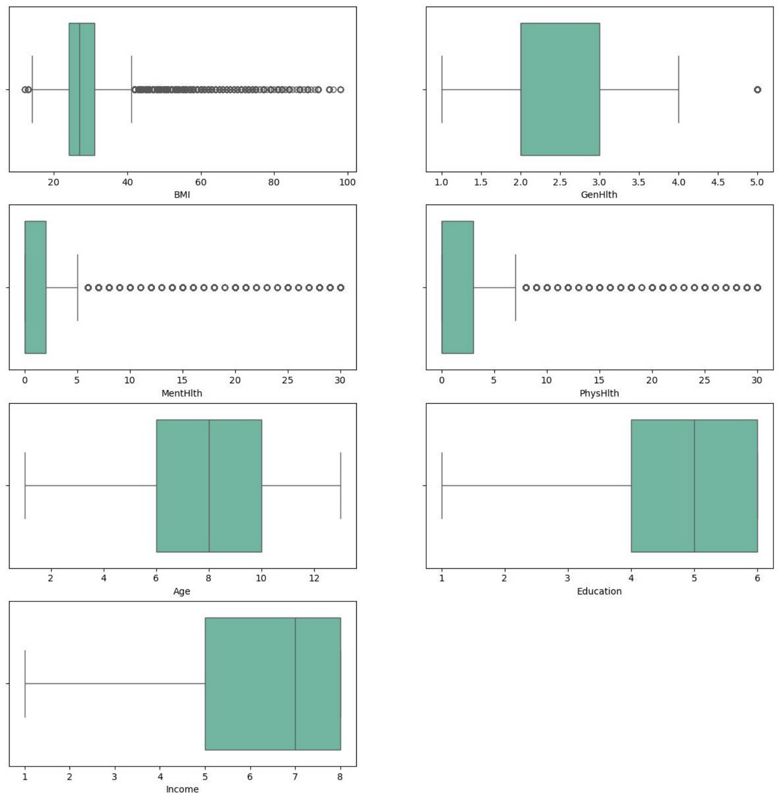


Figure 1: Visualization of Dataset Outliers

It is obvious that quite a lot of outliers are visible in the boxplots of BMI, MentHlth and PhysHlth. However, as these features are all either categorical or discrete numerical with a limited range, the results produced by boxplots is proven to be not indicative of outliers. Therefore, we only need to ensure that no values are unrealistic (eg. falling out of predefined range), which can confuse our models.

3.4.2 Feature Selection

The dataset consists of 21 health indicators, which is quite large because in many situations patients cannot provide that much information. Furthermore, an excessive number of features can contribute to overfitting and increased computational complexity, which limits the practicality of our models in real-world scenarios. Therefore, a strategy to remove irrelevant features and reduce the dimensionality of the dataset is necessary. Before performing feature selection, we need to understand the relationships between features, especially between the independent and the target variable. Pearson Correlation [24] was selected to measure the relation between each pair of features as shown in Figure 2.

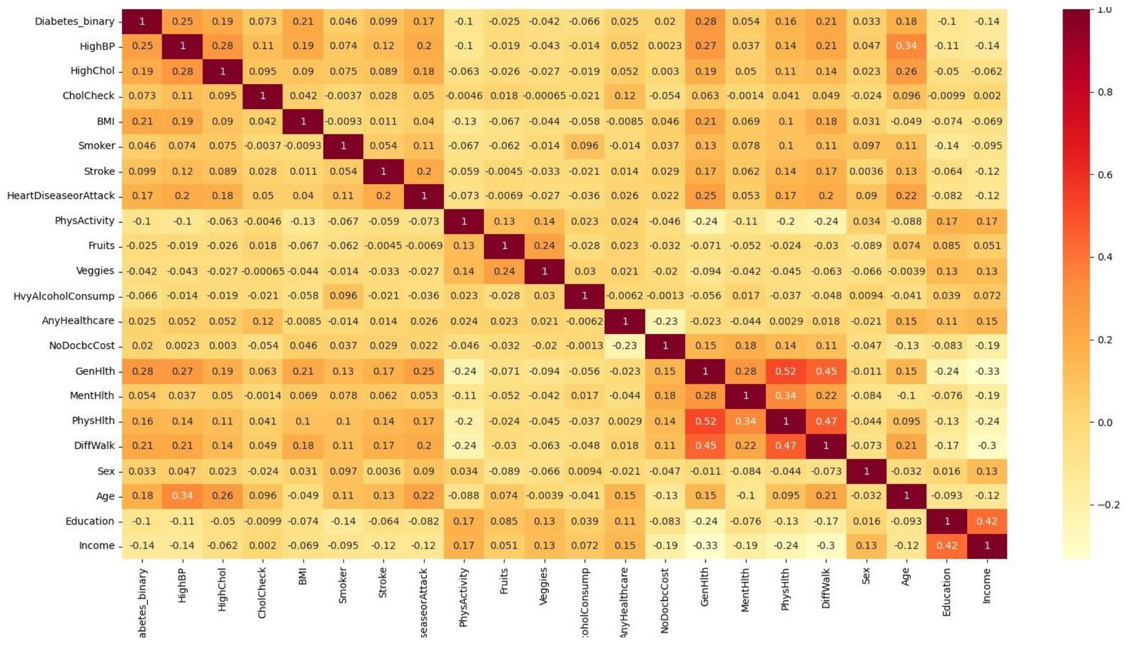


Figure 2: Correlations between each pair of features

From the visualization, it can be inferred that Fruits, AnyHealthcare, NoDocbccost, and Sex are those predictors with lowest correlations with the target variable. The other features are supposed to be quite important in explaining the variance of the output.

We also performed the VIF (Variance Inflation Factor) Test [25] and implemented the Chi-Square [26] method to reliably select important features. The ViF test evaluates multicollinearity in a set of predictors by checking the correlation with other independent variables. Vif can be calculated using the formula below:

$$VIF = \frac{1}{1 - R_j^2} \tag{1}$$

Where R^2 is the coefficient of determination from regressing a predictor variable against all other predictors in the model. The result of the Vif test is shown below.

Index	**Variable**	**VIF**
15	GenHlth	1.821914
17	PhysHlth	1.623288
18	DiffWalk	1.536636
22	Income	1.505649
20	Age	1.354954
2	HighBP	1.344502
21	Education	1.326495
16	MentHlth	1.239497
1	Diabetes_binary	1.193120
3	HighChol	1.180932
8	HeartDiseaseorAttack	1.175776
5	BMI	1.160280
9	PhysActivity	1.157396
14	NoDocbcCost	1.144200
13	AnyHealthcare	1.113209
10	Fruits	1.112540
11	Veggies	1.112397
6	Smoker	1.091872
7	Stroke	1.081612
19	Sex	1.075748
4	CholCheck	1.033501
12	HvyAlcoholConsump	1.025418

Table 3: Variance Inflation Factor (VIF) for Different Variables

A high ViF score means a predictor can easily be predicted by other factors, which makes that feature redundant in the model. Generally, a Vif score over 5 indicates potential multicollinearity, however in this case, all predictors are not highly correlated, thus suggest that multicollinearity may not be a problem.

Chi-Square is a method for examining the associations between each predictor and the target variable by measuring the difference between the expected and the observed counts in the contingency table.

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} \tag{2}$$

Where O_i is the observed value and E_i is the expected value.

The value of Chi-Square score can indicate the importance of a feature, with a higher score is associated with stronger relationship with the target variable.

Index	Feature	Score
15	PhysHlth	133424.406534
14	MentHlth	21029.632228
3	BMI	18355.166400
16	DiffWalk	10059.506391
0	HighBP	10029.013935
13	GenHlth	9938.507776
18	Age	9276.141199
6	HeartDiseaseorAttack	7221.975378
1	HighChol	5859.710582
20	Income	4829.816361
5	Stroke	2725.225194
7	PhysActivity	861.887532
10	HvyAlcoholConsump	779.424807
19	Education	756.035496
4	Smoker	521.978858
12	NoDocbcCost	229.542412
8	Fruits	154.291404
9	Veggies	153.169215
17	Sex	140.248274
2	CholCheck	39.716825
11	AnyHealthcare	3.280938

Table 4: Chi-Square Score for Different Variables

From the result of Chi-Square, we conclude that the top 5 least important features are Fruits, Veggies, Sex, CholCheck and AnyHealthcare, which is closely aligned with the inference from Pearson Correlation [24] plot. These predictors will be removed to enable our models to learn more effectively and reduce the risk of overfitting as well as excessive computational overhead.

3.4.3 Handling Imbalanced Dataset

Furthermore, it is important to address the nature of the dataset being unbalanced, as shown in Figure 3.

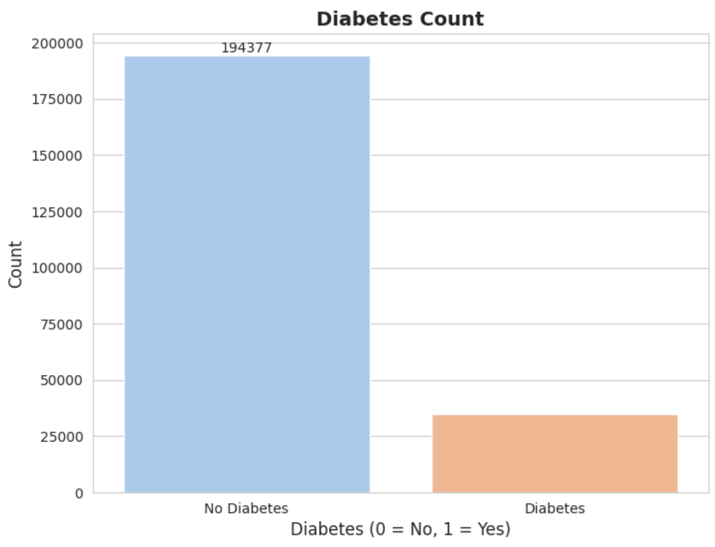


Figure 3: Number of Diabetic and Non-diabetic Records before Undersampling

In this study, we will perform undersampling instead of SMOTE. It will greatly help reduce computational time while still keeping the same or potentially better accuracy. As our classifier becomes well-trained, it tends to create the same class distribution. The undersampled data is shown in the below figure.

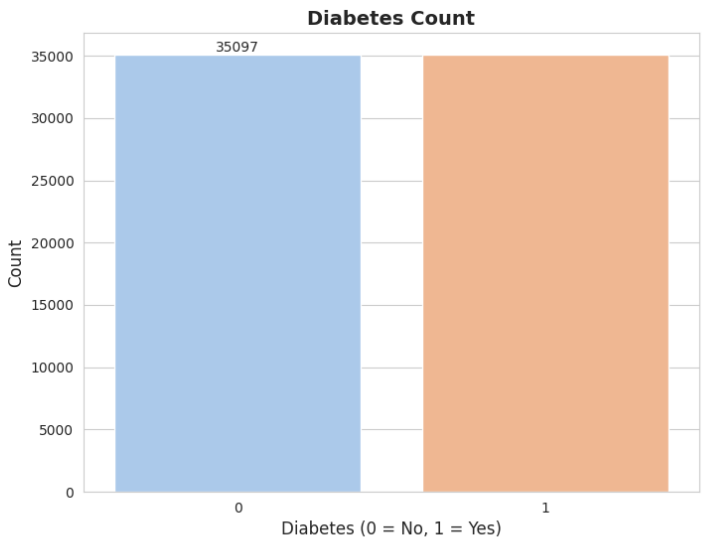


Figure 4: Undersampled Records of Diabetic and Non-diabetic

4 Methods

4.1 Formulation and Implementation of Classification Model

In this work, we evaluate the performance of different optimization techniques of a classification model based on Logistic Regression.

4.1.1 Logistic Regression

Logistic Regression is a fundamental linear classifier that is used for binary classification in this study. Given an input feature vector $\mathbf{x}$, the model predicts the probability that the output $\mathbf{y}$ belongs to class 1. By learning from a training set, Logistic Regression solves this task by making a decision represented as a linear combination of the feature **weights** and **bias**, finally mapping the instance into a probability using a sigmoid function,

$$\sigma(\mathbf{w}^T\mathbf{x} + b) = \frac{1}{1 + \exp\left(-(\mathbf{w}^T\mathbf{x} + b)\right)} \tag{3}$$

where $\sigma(.)$ is the sigmoid function, $\mathbf{w}$ is the weight vector and b is the bias.

We classify an observation x_i by setting a threshold, which is set to 0.5 in this study.

$$\hat{y} = \begin{cases} 1, & \text{if } \sigma(z_i) \geq 0.5, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

We calculate the negative log likelihood loss, generally called **cross-entropy loss**.

$$L_{CE} = -[y \log \sigma(\mathbf{w}^T\mathbf{x} + b) + (1 - y)\log(1 - \sigma(\mathbf{w}^T\mathbf{x} + b))] \tag{5}$$

We train our model by minimizing (5),

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} L_{CE}(\hat{y}^{(i)}, y^{(i)}) \tag{6}$$

where m is the number of training samples, and $y^{(i)}$ represents the actual class label for sample .

4.1.2 Gradient Descent Variants

To minimize the $J(\theta)$ on (6), we use Gradient Descent, one of the most common and famous algorithms to perform optimization. GD optimizes $J(\theta)$ parameterized by a model's parameters θ in the opposite direction of the gradient objective function $\nabla J(\theta)$ w.r.t the parameters [27]. The learning rate η determines how fast we move down the slope until we reach the local minimum.

$$\theta = \theta - \eta \nabla J(\theta) \tag{7}$$

In this study, we analyzed 3 different GD variants : Batch, Mini-Batch and Momentum-based SGD. The momentum-based SGD formula helps accelerate SGD in the relevant direction and dampens oscillations, this is done by having a memory to implement at a moving average.

$$v_t = \gamma v_{t-1} + \eta \nabla J(\theta) \tag{8}$$

$$\theta = \theta - v_t \tag{9}$$

4.1.3 Regularization

As the model learn through training process, it can overfit by capturing noise in the underlying patterns of data, which increases the model's variance. High variance occurs when a small change in the training data can lead to considerable fluctuations in the model's parameters, resulting in poor performance on unseen data. To mitigate this, Ridge Regularization [28] (also known as L2 Regularization) is adopted. This technique limits the model's complexity by putting a constraint on how large its parameters can grow. A penalty term is added to the loss function which balances the trade-offs between training accuracy and model's complexity. The formula for L2 Regularization is as follow:

$$J(\theta) = \sum_{i=1}^{m} \left(y_i - \theta^T \mathbf{x}_i\right)^2 + \lambda \sum_{j=1}^{p} w_j^2$$

Now the model attempts to minimize both the loss term and the regularization term instead of solely optimizing the loss term. This method for limiting the model's complexity has been proven effective especially when the number of features is large.

Another technique for preventing overfitting used in this study is Early Stopping [29]. This method involves a validation set separate from the training set is monitored in the training process. At some point, the training error continues to drop but the validation error stagnates or starts to increase. After the number of epochs in which no considerable drop in validation error is recorded exceeds the predefined patience number, the training process automatically terminates.

4.1.4 Implementation

The model is implemented using Python in Google Colab. By implementing the model as a class, the key parameters are initialized as follow.

```
{
learning_rate = 0.01
epochs = 100
threshold = 0.5
batch_size = 32
method = 'mini_batch'
momentum = 0.9
reg_method = 'l2'
reg_strength = 0.001
alpha = 0.5
patience = 5
}
```

By default, the model uses Mini-Batch Gradient Descent as its default optimizer, which improves convergence by updating weights using small batches of data (32 by default) rather than the entire dataset. Additionally, Momentum-based Stochastic Gradient Descent and Batch Gradient Descent are also implemented for the purpose of this study. The main regularization techniques in this study are L2 Regularization where the regularization parameters *reg_strength* determines how strong it penalize our model and Early Stopping parameterized by a patience meter. We will be iteratively evaluate optimization methods using these default parameters before tuning them to achieve a better result below.

4.2 Training and Testing Methods

4.2.1 Data Splitting

The data is split into training and testing set. A common ratio for training and testing is 80:20. However as ES is also integrated, the data is split into a ratio of 80:10:10 as a training set, validation set and a test set respectively. See Fig.5 for a clear visualization of the proposed training and testing method

4.2.2 Normalization

As previously stated in our work, we intend to design a model based on the LR algorithm incorporated with different optimizers for further analysis. Therefore, we performed scaling for all predictors by normalizing the data into the range of 0 to 1. This would enhance our model performance as it prevents computation overflow and uneven penalization for features with large magnitudes. Furthermore, by normalizing data, our optimization algorithm would be stable.

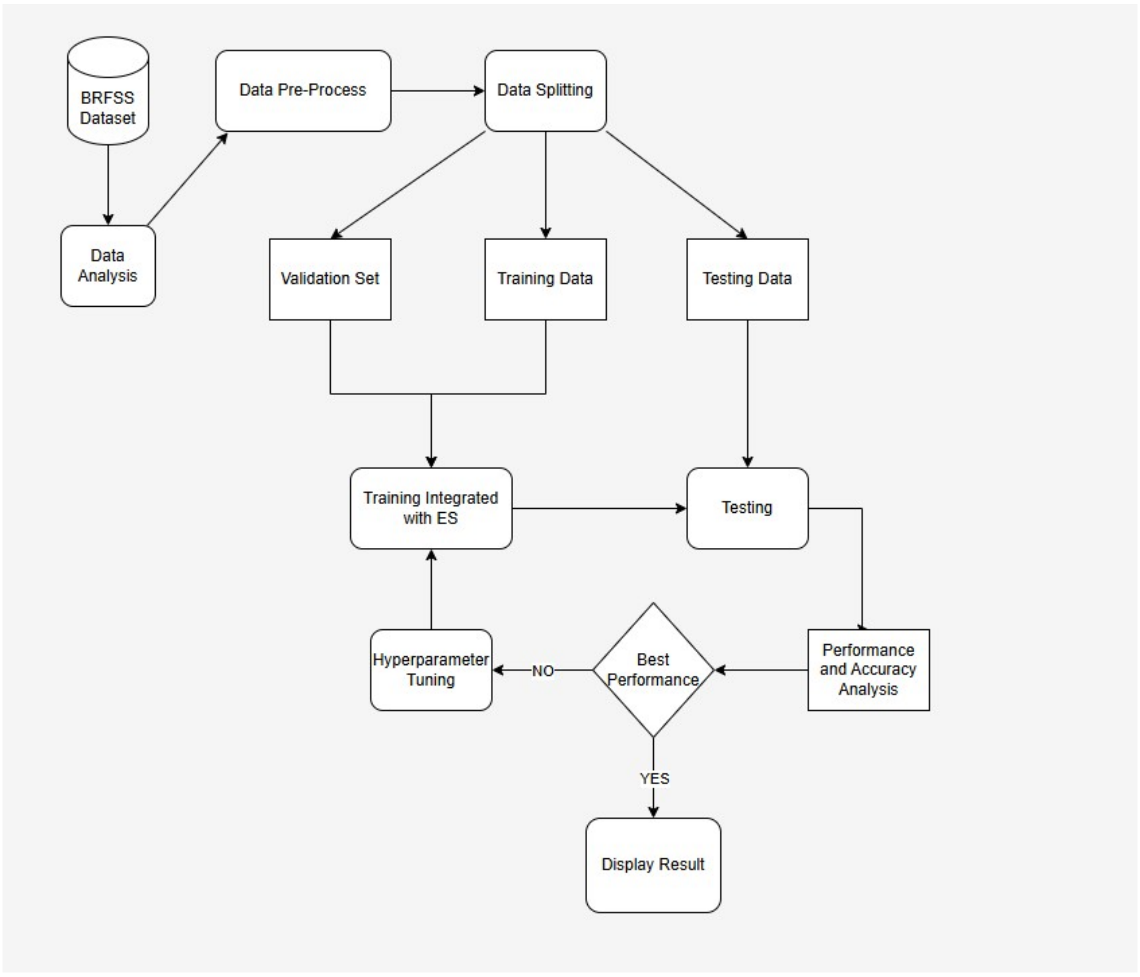


Figure 5: Proposed Training and Testing Method

4.3 Results

Comparative analysis is done by different metrics, calculated using four important terms:- True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). TP and TN represent observations where the result and the actual outcome aligns.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$Precision \frac{TP}{TP + FP} \tag{12}$$

$$F_1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{13}$$

A classification report is generated regarding Accuracy, Recall, Precision and F_1-Score (see Eqs. 10-13). Recall refers to the percentage of positive results being correctly identified. Precision shows what percent of predictions are correct. The F_1-Score is the percentage of positive predictions that are correct.

Optimizer	Precision	Recall	F1-Score	Support
Batch GD				
0	0.71	0.72	0.72	3510
1	0.72	0.71	0.71	3509
Accuracy		0.72		7019
Macro Avg	0.72	0.72	0.72	7019
Weighted Avg	0.72	0.72	0.72	7019
Mini-Batch GD				
0	0.74	0.71	0.72	3510
1	0.72	0.75	0.74	3509
Accuracy		0.73		7019
Macro Avg	0.73	0.73	0.73	7019
Weighted Avg	0.73	0.73	0.73	7019
Momentum-SGD				
0	0.77	0.57	0.66	3510
1	0.67	0.85	0.75	3509
Accuracy		0.69		7019
Macro Avg	0.73	0.71	0.71	7019
Weighted Avg	0.73	0.71	0.71	7019

Table 5: Classification Report for Batch, Mini-Batch, and SGD (Before Tuning)

Table 5 shows the classification report for every optimization algorithm. Where all three optimizers performs generally well on this dataset.

4.4 Improving Performance

4.4.1 Hyperparameter Tuning

Tuning was carried out by using simple grid search, where we trained the model with every possible combination of parameters while setting the epochs at 1000 until we find the best combination of a certain optimizer. Table 6 shows the best set of parameters and accuracy for each optimizer.

Method	Learning Rate	Reg. Strength (L2)	Batch Size	Momentum	Score
Batch GD	0.01	0.005			0.73
Mini-Batch GD	0.01	0.005	32		0.73
MSGD	0.01	0.005		0.5	0.73

Table 6: Hyperparameters Tuning Result (1000 epochs)

4.4.2 Early Stopping

We then proceed to optimize our runtime by implementing Early Stopping (ES). Multiple tests are conducted using various choices of the threshold, results for each optimizer are shown below on Table 7.

Method	Patience	Epochs for convergence
Batch GD	5	600
Mini-Batch GD	15	60-100 (Due to shuffling)
MSGD	15	30-50 (Due to shuffling)

Table 7: Best Threshold for ES

The effect of ES can be seen improving the performance of all optimizers on Table 8. Accuracies were also enhanced as the result of hyperparameters tuning.

Method	Accuracy	Loss	Time
Batch (Before)	0.72	0.60	9.5565
Batch (After)	0.73	0.55	6.7626
Mini-Batch (Before)	0.73	0.54	7.6265
Mini-Batch (After)	0.73	0.55	2.2419
SGD (Before)	0.69	0.67	98.0939
SGD (After)	0.73	0.56	13.9100

Table 8: Comparison of Accuracy, Loss, and Time Before and After applying ES and Tuning

4.5 Summary

Our model was first designed based on LR and three optimizers, trained and tested with untuned parameters (see Table 5). Hyperparameter tuning and ES were later employed to enhance accuracy while optimizing execution time. Table 9 and Figure 6 show the confusion matrices and reports after applying ES and Tuning.

Method	**Precision**	**Recall**	**F1-Score**	**Support**
Batch GD				
Class 0	0.73	0.71	0.72	3510
Class 1	0.72	0.74	0.73	3509
Accuracy		0.73		7019
Macro Avg	0.73	0.73	0.73	7019
Weighted Avg	0.73	0.73	0.73	7019
Mini-Batch GD				
Class 0	0.74	0.71	0.72	3510
Class 1	0.72	0.75	0.74	3509
Accuracy		0.73		7019
Macro Avg	0.73	0.73	0.73	7019
Weighted Avg	0.73	0.73	0.73	7019
MSGD				
Class 0	0.72	0.73	0.72	3510
Class 1	0.73	0.72	0.72	3509
Accuracy		0.72		7019
Macro Avg	0.72	0.72	0.72	7019
Weighted Avg	0.72	0.72	0.72	7019

Table 9: Classification report for Batch, Mini-Batch, and MSGD with tuning and ES

Batch GD

True	**Predicted**	
	0	**1**
0	2479 (0.71)	1031 (0.29)
1	898 (0.26)	2611 (0.74)

Mini-Batch GD

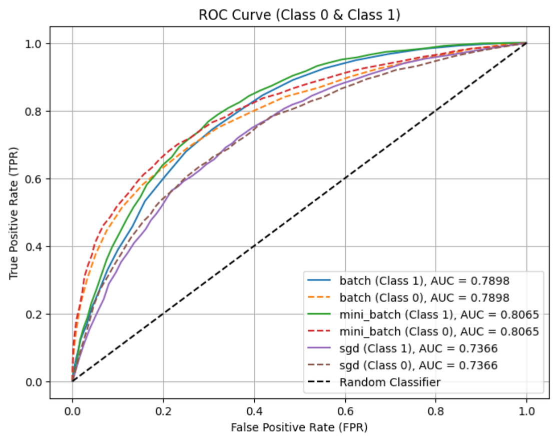
True	**Predicted**	
	0	**1**
0	2478 (0.71)	1023 (0.29)
1	872 (0.25)	2637 (0.75)

MSGD

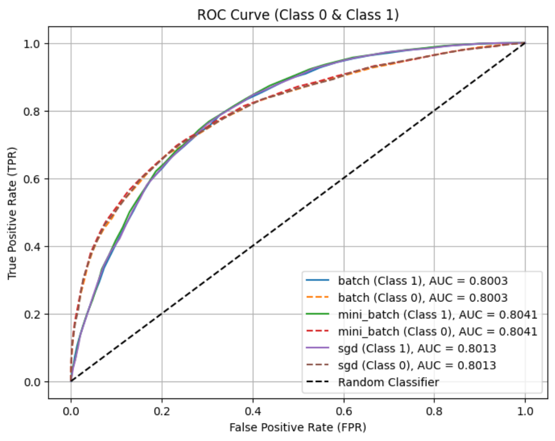
True	**Predicted**	
	0	**1**
0	2444 (0.70)	1066 (0.30)
1	853 (0.24)	2656 (0.76)

Figure 6: Confusion matrices showing classification performance across different gradient descent optimization methods: Batch Gradient Descent (Batch GD), Mini-Batch Gradient Descent (Mini-Batch GD), and Stochastic Gradient Descent (SGD).

Furthermore, Figure 7 shows the comparative ROC Curve between optimizers (Untuned and Tuned). Both curves suggests that the Mini-Batch GD (batch size 32) perform better both untuned and tuned. Figure 7b shows that all optimizers work relatively well for this study.



(a) ROC Curve



(b) ROC Curve (Peformance Enhancing Methods)

Figure 7: Comparison of ROC Curves

5 Conclusion

In this paper, we have proposed an optimized approach for predicting diabetes mellitus using Logistic Regression configured with different hyper-parameters and Early Stopping. Through rigorous testing and tuning, we concluded that the Mini-batch optimizer integrated with Early Stopping provides a balanced tradeoff between execution speed and accuracy. Although Logistic Regression is a basic machine learning algorithm, the performance of our models on the BRFSS2015 dataset is efficient enough to be applicable to real-life settings. By adopting different optimization methods, we were able to achieve better performance than the built-in methods provided by Scikit-Learn like LogisticRegression or SGDClassifier. Although the accuracy of our models is not comparable with those proposed by the authors of [17], our approach can perform equally well between two classes, Diabetic and Non-Diabetic. In the future, we will explore more machine learning methods to achieve a better result in diabetes prediction, making a contribution to the fight against this chronic disease.

References

[1] World Health Organization. Diabetes fact sheet, 2024.

[2] Medical News Today. What to know about insulin, 2019.

[3] WebMD. Diabetes causes, 2024.

[4] Mayo Clinic. Prediabetes - diagnosis & treatment, 2024.

[5] Diabetes.co.uk. Blood sugar level ranges, 2024.

[6] Healthgrades. Is there a cure for diabetes?, 2024.

[7] Better Health Channel. Diabetes - long-term effects, 2024.

[8] Seema Abhijeet Kaveeshwar and Jon Cornwall. The current state of diabetes mellitus in india. *The Australasian medical journal*, 7(1):45, 2014.

[9] Centers for Disease Control and Prevention (CDC). What is the national dpp?, 2024.

[10] G Swapna, R Vinayakumar, and KP Soman. Diabetes detection using deep learning algorithms. *ICT express*, 4(4):243–246, 2018.

[11] Mark W Craven and Jude W Shavlik. Using neural networks for data mining. *Future generation computer systems*, 13(2-3):211–229, 1997.

[12] NVIDIA. What's the difference between artificial intelligence, machine learning, and deep learning?, 2024.

[13] Aishwarya Mujumdar and Vb Vaidehi. Diabetes prediction using machine learning algorithms. *Procedia Computer Science*, 165:292–299, 2019.

[14] Jobeda Jamal Khanam and Simon Y Foo. A comparison of machine learning algorithms for diabetes prediction. *Ict Express*, 7(4):432–439, 2021.

[15] Ram D Joshi and Chandra K Dhakal. Predicting type 2 diabetes using logistic regression and machine learning approaches. *International journal of environmental research and public health*, 18(14):7346, 2021.

[16] Priyanka Rajendra and Shahram Latifi. Prediction of diabetes using logistic regression and ensemble techniques. *Computer Methods and Programs in Biomedicine Update*, 1:100032, 2021.

[17] MARWA HUSSEIN MOHAMED, MOHAMED HELMY KHAFAGY, NESMA MOHAMED, MAHMOUD KAMEL, and WAEL SAID. Diabetic mellitus prediction with brfss data sets. *Journal of Theoretical and Applied Information Technology*, 102(3), 2024.

[18] Centers for Disease Control and Prevention. Behavioral risk factor surveillance system (brfss), 2025. Accessed: 15-Mar-2025.

[19] Zidian Xie, Olga Nikolayeva, Jiebo Luo, and Dongmei Li. Building risk prediction models for type 2 diabetes using machine learning techniques. *Preventing chronic disease*, 16:E130, 2019.

[20] David W Brown, Lina S Balluz, Wayne H Giles, Gloria L Beckles, David G Moriarty, Earl S Ford, and Ali H Mokdad. Diabetes mellitus and health-related quality of life among older adults: Findings from the behavioral risk factor

surveillance system (brfss). *Diabetes research and clinical practice*, 65(2):105–115, 2004.

[21] Mohammad Mihrab Chowdhury, Ragib Shahariar Ayon, and Md Sakhawat Hossain. An investigation of machine learning algorithms and data augmentation techniques for diabetes diagnosis using class imbalanced brfss dataset. *Healthcare Analytics*, 5:100297, 2024.

[22] Alex Teboul. Diabetes health indicators dataset.

[23] Centers for Disease Control and Prevention (CDC). Behavioral risk factor surveillance system (brfss) 2015 annual data, 2015. Accessed: 2025-03-15.

[24] Karl Pearson. Vii. mathematical contributions to the theory of evolution.—iii. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, (187):253–318, 1896.

[25] David A Belsley, Edwin Kuh, and Roy E Welsch. *Regression diagnostics: Identifying influential data and sources of collinearity.* John Wiley & Sons, 2005.

[26] Karl Pearson. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

[27] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[28] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[29] Anders Krogh and John Hertz. A simple weight decay can improve generalization. *Advances in neural information processing systems*, 4, 1991.