CSC 5825

**LOGISTIC REGRESSION IMPLEMENTATION**

INTRODUCTION TO MACHINE LEARNING

HOMEWORK 1



Chart

Description automatically generated with medium confidence

**Prepared By: Anika Tasnim (go0273)  
Date: 09/21/2022**

**Data Attribute Info:**

Logistic Regression is a statistical analysis method which is used when the dependent variable is binary or can have only two outputs. Here are the features of the dataset:

* Age: age of the patient [years]
* Sex: sex of the patient [M: Male, F: Female]
* ChestPainType: chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
* RestingBP: resting blood pressure [mm Hg]
* Cholesterol: serum cholesterol [mm/dl]
* FastingBS: fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]
* RestingECG: resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria]
* MaxHR: maximum heart rate achieved [Numeric value between 60 and 202]
* ExerciseAngina: exercise-induced angina [Y: Yes, N: No]
* Oldpeak: oldpeak = ST [Numeric value measured in depression]
* ST\_Slope: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, down: downsloping]
* HeartDisease: output class [1: heart disease, 0: Normal]

Before building the model, we check if there are any missing values or not. Then we check if there are any duplicate rows and the general description of the dataset. As there are no missing values or duplicate rows in the dataset and the number of patient’s heartdisease is fairly distributed we proceed to do feature engineering next step.

**Data preprocessing & Feature Engineering:**For this project we have only used one hot encoding for the categorical features. Then we split the data into 80%-20% train-split dataset. Then finally normalization of the training and testing dataset separately to scale the data.

**Logistic Regression:**

## ****Logistic regression is also used to estimate the relationship between a dependent variable and one or more independent variables, but it is used to make a prediction about a categorical variable versus a continuous one.****

## ****The steps to build a custom Logistic Regression Algorithm:****

• Define the model structure (data shape).  
• Initialize model parameters (W, B).  
• Learn the parameters for the model by minimizing the cost:  
  - Calculate current loss (forward propagation). Apply the linear and then nonlinear transformation (Sigmoid Function) on the input features   
  - Calculate current gradient (backward propagation). Deriving the first derivatives of the loss function with respect to the weights (W) and bias (B)  
  - Update parameters (gradient descent).  
• Use the learned parameters to make predictions (on the test set).  
• Analyze the results and find the accuracy of the model.

**Forward Propagation:**

The X is passed through a linear model that contains the weights (W) and bias (B). The linear combination of the inputs is then multiplied by an activation function (sigmoid). In this process, weights and biases are propagated from inputs to output is called forward propagation. After arriving at the predicted output, the loss for the training example is calculated.

**Diagram, schematic

Description automatically generated**

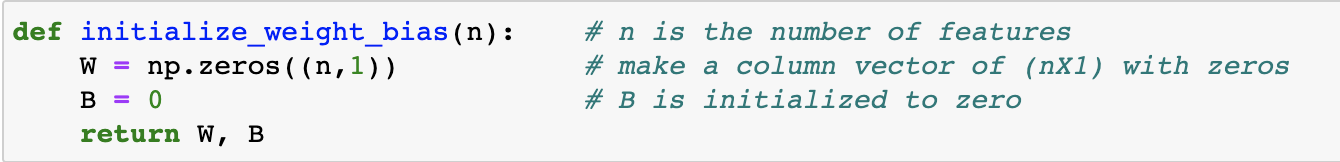
**The final cost function for binary classification:**

**Text

Description automatically generated with medium confidence**

**Initialization of Parameters W & B**

The data sets are always multidimensional. We will need to use matrices for any kind of calculation. So, for input, we have two matrices to deal with. The first one is for feature vectors, and the second is for parameters or weights. Our first matrix is of the m X n dimension, where m is the number of observations while n is the dimension of observations. And the second one is of nx1 dimension. Here, we will add a bias (0) initially to our feature vectors matrix and a corresponding parameter term to the weight vector. Bias is important to make the model more flexible.

****

**Sigmoid Function:**

In a linear regression model, the hypothesis function is a linear combination of parameters given as y = ax+b for a simple single parameter data. This allows us to predict continuous values effectively, but in logistic regression, the response variables are binomial, either ‘yes’ or ‘no’. So, it makes less sense to use the linear function to predict anything except the values between 0 and 1. And the most effective function to limit the results of a linear equation to [0,1] is the sigmoid or logistic function.

**Diagram

Description automatically generated with low confidence**

**Backward Propagation:**

To train our model we need to update the weights and biases that were initialized to zero. Therefore, we need to find the derivatives of the loss function with respect to W and B i.e., dJ/dW and dJ/dB.

By applying the chain rule:

X = input matrix

Z = W.X+B

A = sigmoid(Z)

dJ/dW = (dJ/dZ)(dZ/dW) = (A-Y)XT

dJ/dB = (dJ/dZ)(dZ/dB) = (A-Y)

then:

W = W – learning rate \* dJ/dW

B = B – learning rate \* dJ/dB

**Accuracy:  
85.33%**

**Cross Validation (5 folds & 10 folds)**

1. The dataset is split into training and test dataset.
2. The training dataset is then split into K-folds. (5 in this project)
3. Out of the K-folds, (K-1) fold is used for training
4. 1 fold is used for testing
5. The model with specific hyperparameters is trained with training data (K-1 folds) and validation data as 1 fold. The performance of the model is recorded.
6. The above steps (step 3, step 4, and step 5) is repeated until each of the k-fold got used for validation purpose. This is why it is called k-fold cross-validation.
7. Finally, the mean and standard deviation of the model performance is computed by taking all of the model scores calculated in step 5 for each of the K models.
8. Step 3 to Step 7 is repeated for different values of hyperparameters.
9. Finally, the hyperparameters which result in the most optimal mean and the standard value of model scores get selected.
10. The model is then trained using the training data set (step 2) and the model performance is computed on the test data set (step 1).

**2. Confusion Matrix**

The confusion matrix detects the count of **TP** (True Positive), **TN** (True Negative), **FP (**False Positive), **FN** (False Negative) in the predictions of a classifier.

From Confusion matrix we can derive the **Accuracy** which is given by **the sum of the corrected predictions** divided by **the total number of predictions**:

* **Precision = TP / TP + FP** or **Positive Predicted Value**
* **Sensitivity** or **Recall = TP / TP + FN** or **True-Positive-Rate (TPR)**
* **Accuracy = TP+TN/TP+FP+FN+TN**
* F1 **Score = 2\*(Recall \* Precision) / (Recall + Precision)**

For good classifiers, **TPR and TNR** both should be nearer to **100%**. Similar is the case with **precision** and **accuracy** parameters. On the contrary, **FPR and FNR** both should be as close to **0%** as possible.

Conclusion:

The accuracy is improved with cross validation, and the custom Logistic regression model performs slightly better than the built-in library function. These are the following results from the code:

The accuracy using the library function is 84.24%.

The F1 score using the library function is 84%

The accuracy using the library function and cross validation library is 86.714%

The accuracy using the custom Logistic Regression Function is 85.33%

The F1 score generated from the custom LR function is 86.96%

The accuracy using the custom Logistic Regression Function and applying cross validation is 86.6%