Assignment: 2

Name: Uday Kumar Kamalapuram
GMU Id: G01340201
Miner User Id: uday
Public Score: 0.91
Rank: 21

APPROACH

Selecting the Notebook:

I have chosen Google colab to start the Assignment as I am familiar with Google colab and I can work on it from the cloud.

Data Uploading and Framing:

I have imported the given files "1644871288_9762487_cleveland-train.csv" and "1644871288_9775174_cleveland-test.csv" through files.upload() from google.colab library. I am reading the files through Pandas library read_csv() and storing them as pandas data frame with columns as there in csv, except I am renaming the "heartdisease::category|-1|1" column to "res" for easy naming convention.

Data Pre-processing:

I have tried different approaches in data processing, first I have converted all -1 elements in "res" column to 0 to get the formula as present in Text books and other resources in Internet. I have tried standardizing the data with StandardScaler.fit_transform method, I got the accuracies as shown in **Table 1**. In another approach I tried to convert categorical values ("cp","thal","slope") into binary categorical values by using pandas.get_dummies() function, for that data I got accuracies as shown in **Table 2.**

Train-Test Split:

I have split the trained data using train_test_split from sklearn.model_selection. I have used this to get the accuracy of the model before uploading it to Miner. I have checked the accuracy on the train data by splitting the train data to x_test, x_train, y_train, y_test.

Building Model:

First, I have written the method for sigmoid function then I have written method "gradientCalculation()" to calculate hypothesis of logistic regression "hθ(x)" and cross entropy cost "J(θ)" and gradient weights and bias and storing gradient weights and bias in "gradients" dictionary, I am returning the cost and "gradients" dictionary from this function.

$$h\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}} \qquad J(\theta) = -\frac{1}{m}\sum\left[y^{(i)}\log(h\theta(x(i))) + \left(1 - y^{(i)}\right)\log(1 - h\theta(x(i)))\right] \qquad \theta j := \theta j - \alpha\frac{\partial}{\partial \theta j}J(\theta)$$

I written "updateWeights()"function to update the weights "θj" for each iteration and for each iteration I am calling gradientCalculation() method. After all iteration I am returning the final weights and bias and plotting the cross-entropy cost function. I also placed termination condition in the function when each of the gradient is less than 10^-3, I will terminate the iteration. Finally I am calling the "logistic_regression_training()" function where I am passing the x_train, x_test, y_train, y_test, learningRate and iterations and training the model and getting weights and bias and sing those to predict the heart disease for the test data. I written the predicted values into a file and uploaded to miner website.

Learning Rate	Iterations	Training Model Accuracy	Miner Accuracy	Cross Entropy Cost Function	Classification Error on Training Data	Classification Error on Test Data (Miner)	Time taken to Train Model
10,000	10^-5	0.84	0.91	0.6662	0.16	0.09	989 ms
100,000	10^-5	0.84	0.91	0.5343	0.16	0.09	6.66 s
1,000,000	10^-5	0.80	0.91	0.4283	0.19	0.09	1min 5s

Table 1:

Model with Categorized value

Assignment: 2

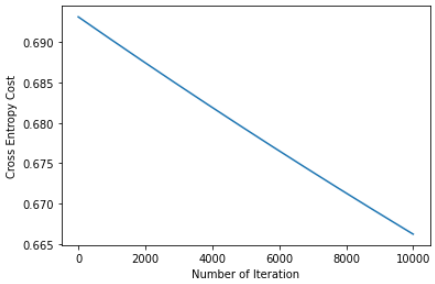
Learning Rate	Iterations	Training Model Accuracy	Miner Accuracy	Cross Entropy Cost Function	Classification Error on Training Data	Classification Error on Test Data (Miner)	Time taken to Train Model
10,000	10^-5	0.94	0.91	0.6567	0.06	0.09	1.58 s
100,000	10^-5	0.94	0.90	0.5093	0.06	0.1	6.7 s
1,000,000	10^-5	0.87	0.90	0.4283	0.13	0.1	1min 3s

Table 2: Model converted Categorized values to binary.

Observations:

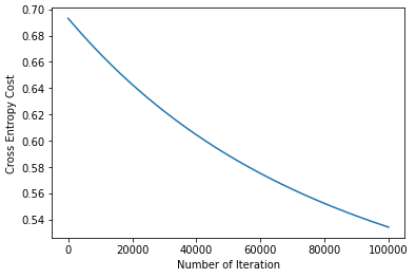
Iteration: 10,000 Learning Rate: 10^-5

While training the data with learning rate 10^-5 and iterations 10,000. I have observed the cross-entropy cost function of 0.6662 and the classification error on the training data is 0.16, The estimated classification error on the test data from miner is 0.09, from this classification error difference the model has good generalization. The time taken to train model is 989 ms.



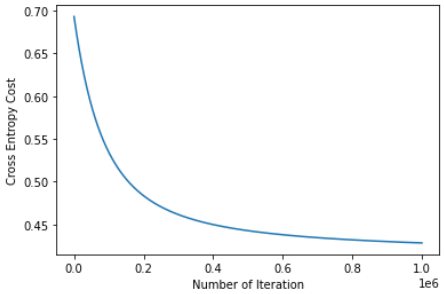
Iteration: 100,000 Learning Rate: 10^-5

While training the data with learning rate 10^-5 and iterations 100,000. I have observed the cross-entropy cost function of 0. 5343 and the classification error on the training data is 0.16, The estimated classification error on the test data from miner is 0.09, from this classification error difference the model has adapted good generalization. The time taken to train model is 6.66 s.



Iteration: 1,000,000 Learning Rate: 10^-5

While training the data with learning rate 10^-5 and iterations 100,000. I have observed the cross-entropy cost function of 0. 5343 and the classification error on the training data is 0.2, The estimated classification error on the test data from miner is 0.09, from this classification error difference the model has good generalization. The time taken to train model is 1min 5s.



Sklearn Library LogisticRegression:

I have imported LogisticRegression from sklearn.linear_model. I used the fit method to train the model with the x_train and y_train data which I generated from the train-test split. After that I predicted the output for x_test and compared with the y_test. I got 0.8065 accuracy for the train-test Split data.

StandardScaler Standardization: I have standardize the data with sklearn StandardScaler().fit_transform() method which has given me better classification rate compare to Min_Max normalization.

Min_Max Normalization: I have normalize the data with "preprocessing.MinMaxScaler()" of sklearn library, in this normalization compare to StandardScaler the accuracy of the model got reduced. For Min_Max normalized data trained model accuracy is 0.74.

References:

https://medium.com/@cdabakoglu/heart-disease-logistic-regression-machine-learning-d0ebf08e55c0
https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148
https://scikit-learn.org/