Miner Username: Michael_Scarn

Loading and Preparing Data: I've used pandas to read data from train CSV using 'pd.read_csv' method. Next Step I have split the data using 'train_test_split' function from 'sklearn.model_selection' to get train and test sets so that we can calculate the classification error on Train Data set better.

Gradient used to Compute for Logistic Regression:

$$\nabla_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i \mathbf{x}_i}{1 + e^{y_i \mathbf{w}\cdot\mathbf{x}_i}}$$

I have expressed the above equation as

' gradient=-(1/y.shape[0])*np.sum((X*y[:, np.newaxis])*(1.0 / (1 + np.exp(y*(np.dot(X,w)))))[:, np.newaxis],axis=0) '

Cross Entropy Error*: cross_entropy=(1/y.shape[0])*np.sum(np.log(1+np.exp(-(y*(np.dot(X,w*)))))))* which is equivalent to

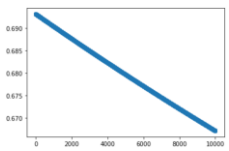
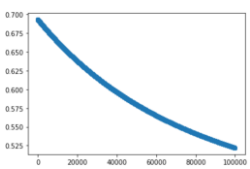
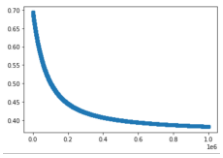
$$E_{\text{in}}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\ln\frac{1}{\sigma(y_i \mathbf{w}\cdot\mathbf{x}_i)} = \frac{1}{n}\sum_{i=1}^{n}\ln(1 + e^{-y_i \mathbf{w}\cdot\mathbf{x}_i})$$

Training using Gradient Descent Algorithm without Scaling/Normalization:

The above table shows the Cross-Entropy Error on Train Data, Classification error on Train Data ,estimate of classification error on test data obtained from miner and the time it took train my model.

Iterations	Learning Rate	Cross-Entropy Error on Train Data	Classification error on Train	Classification Error on Test Data	Time to Train the model
10000	10^{-5}	0.58	0.31	0.29	5s
100000	10^{-5}	0.45	0.23	0.17	11s
1000000	10^{-5}	0.42	0.18	0.14	55s

My model starts by augmenting *intercept/bias = 1* to the X_train vector and initializing *weights = zero* at the beginning of starting iteration. Then for each iteration gradient is computed based on the above equation and weights are updated using '*w -= eta * gradient*' where eta is the learning rate [eta here is 10^{-5}, and the algorithm terminates automatically when each term in the gradient is below 10^{-3} at any step].And the cross-entropy error is calculated based on the above equation. By Varying the iterations using three different bounds on the maximum number of iterations: ten thousand, one hundred thousand, and one million I've observed that as number of iterations increase the cross-entropy error decreases along with the classification error for both train and test sets. However, the I've seen varying differences in my test classification and train classification error. The classification error on test data from miner was less than classification error on train data.(Could be because test has lesser data and miner random samples the distribution/the algorithm generalizes well). The runtime increases as we increase our iterations. I've also observed as the number of iterations increase for the same eta value the curve on the graph plotted for X-axis=iterations, Y-Axis= Cross Entropy Error was less narrow.

The above graphs are for 10k,100k,1 million iterations for eta=10^{-5}.

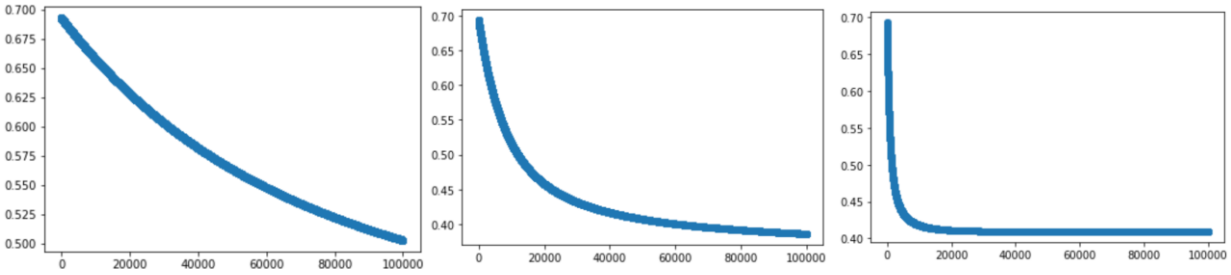
Training using Gradient Descent Algorithm with Scaling/Normalization: Experiments with different iterations and learning rate with tolerance of 10^{-6}

Iterations	Learning Rate	Cross-Entropy error on Train Data	Classification error on Train	Classification error on Test Data	Time to Train the model	Iteration at which train stops for 10^{-6} tolerance
1000000	10^{-5}	0.40	0.14	0.10	40s	1000000
100000	10^{-5}	0.52	0.19	0.11	6s	100000
10000	10^{-5}	0.66	0.21	0.13	3s	10000
100000	10^{-2}	0.34	0.28	0.12	2s	34499
1000000	10^{-3}	0.31	0.28	0.11	33s	326800
100000	10^{-4}	0.42	0.10	0.10	4s	100000
10000	10^{-4}	0.52	0.21	0.14	2s	10000
1000000	10^{-4}	0.41	0.13	0.10	37s	1000000

For each of the features in training data, after subtracting the mean and dividing by the standard deviation the time to train the model has decreased and the gradient descent converges faster. The Classification error was lesser than before scaling the features. With respect to learning rates if eta is small it took slightly more time to converge. With high learning rates the gradient converged faster and reached the minimum quickly as the step size is bigger. For example, with eta=10^{-3} and iterations=1000000 the gradient terms fell below 10^{-6} and algorithm terminated at 326800 iteration. As eta value increased the cross-entropy error decreased because the step size in the direction of negative gradient was bigger than with smaller eta values. With the increase in iterations for the same eta values I was able to achieve better classification accuracy and lesser cross-entropy error values.

Note: To predict the values I have used *' predict_value=1 / (1 + np.exp(-(np.dot(X,w))))'* to calculate the probabilistic value. Then by using a threshold of 0.5 I have classified into 1/-1.

Below graphs are for fixed iterations = 100000 and eta =10^{-5},10^{-4},10^{-3}



Comparison with Logistic Regression model from homework 1:

Classification error on Train	Classification on Test Data	Time to Train the model
0.16	0.13	3s

I have *used 'model = LogisticRegression(C = 1.0,random_state=0)'* using *'sklearn.linear_model'* package

The time taken to train this model was less than with using gradient descent with 100k and 1mil iterations. But the classification error after using gradient descent on 1mil with an eta value of 10^{-5} was lesser than using than *sklearn* 's Logistic Regression model from homework 1.

References :

https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3
https://numpy.org/doc/