Data Mining Project 2: Gradient Descent

Lalithanjana Kollipara (G01386376)
Mail ID: lkollipa@gmu.edu
Miner Username: nocturnal

AIM:
To implement an optimization method (gradient descent) that allows you to train logistic regression models. To better understand the relationship between the optimization objective (cross-entropy error), training set error, and test set error, and how these relate to generalization. To understand the impact of parameters and design choices on the convergence and performance of the gradient descent algorithm and the learned models.

RESULTS AND OBERVATIONS:
Task 1: To analyse the results of the Gradient Descent function for different iterations (10k, 100k, 1M) by keeping the learning rate constant (10^{-5}) on both training and testing data before scaling the training data.

Results of the Gradient Descent function for 10k iterations at a learning rate of 10^{-5}

Iterations (10000)	**Accuracy**	**Classification Error**	**Cross-Entropy Error**
Train Data	0.78	0.22	0.6098
Test Data	0.86	0.14 [estimate]	Time taken by Model: 16.2 s

Results of the Gradient Descent function for 100k iterations at a learning rate of 10^{-5}

Iterations (100000)	**Accuracy**	**Classification Error**	**Cross-Entropy Error**
Train Data	0.80	0.20	0.4639
Test Data	0.85	0.15 [estimate]	Time taken by Model: 2 mins 45 s

Results of the Gradient Descent function for 1M iterations at a learning rate of 10^{-5}

Iterations (1000000)	**Accuracy**	**Classification Error**	**Cross-Entropy Error**
Train Data	0.83	0.17	0.4261
Test Data	0.87	0.13 [estimate]	Time taken by Model: 19 mins 40 s

- From the above observations, we can conclude that the accuracy increases as we increase the number of iterations for the gradient descent function. This is because, the more the number of iterations, the more the model will learn from the training data and we get more accurate results.
- The classification error of the training data gradually decreases with an increase in the number of iterations as the model learns more and more about the training data.
- We also observe that the cross-entropy error decreases with an increase in the number of iterations making the gradient descent converge more effectively.
- The time taken to train the model on the training data increases as the number of iterations increase since the model is learning again and again on the training data.

Task 2: To compare the results of the above model and when the model is trained using the Logistic Regression model from the sklearn library.

Results on training the model using Scikit Learn

	Accuracy	**Classification Error**	**Time for training**
Training without sklearn	0.79	0.21	18.6 s
Training using sklearn	0.82	0.18	332 ms

- From the above table, we can conclude that my gradient descent model is less accurate than the model that is implemented using the already existing sklearn library.
- We also observe that the time taken to train on the training data using the sklearn library is less than the time taken to train the model using my gradient descent function.

Task 3: To analyse the results of the Gradient Descent function for different learning rates (10^{-3}, 10^{-4}, 10^{-5}) by keeping the number of iterations constant (1M) on the training data after scaling the training data.

Results of the Gradient Descent function for 1M iterations after scaling

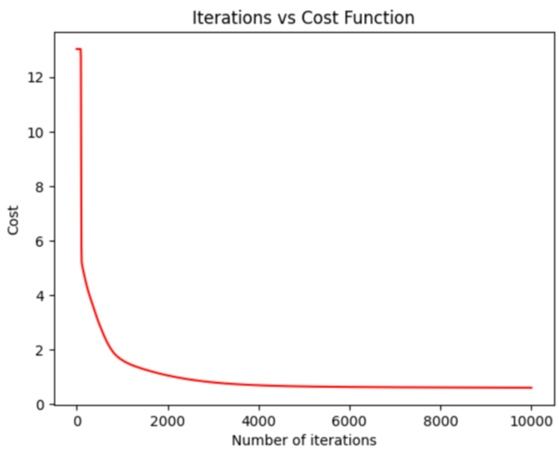
Number of Iterations	**Learning Rate**	**Accuracy**	**Classification Error**	**Cross Entropy Error**	**Time for training**
1000000	10^{-3}	0.81	0.19	0.4147	7.44 s
1000000	10^{-4}	0.81	0.19	0.4147	1 min 3 s
1000000	10^{-5}	0.80	0.20	0.4147	6 min 17 s

- We observe that the accuracy of the model has increased after scaling the features and the time to train the model has also been significantly reduced.
- We can also observe that the cross-entropy error is less when compared to that of before scaling, therefore making the gradient descent converge more effectively.

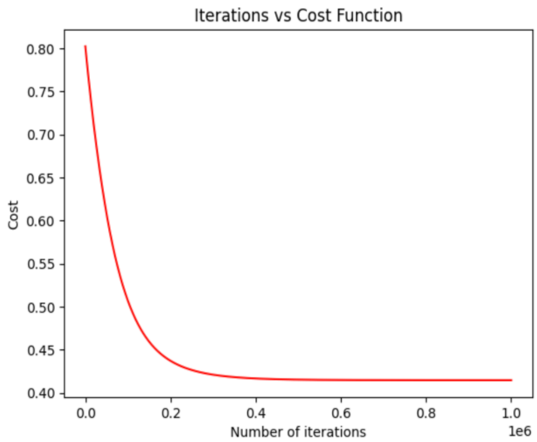
Accuracy and Classification Error before and after Scaling the Features

	Accuracy	**Classification Error**
Before Scaling	0.78	0.22
After Scaling	0.81	0.19

CONVERGENCE PLOTS:



Gradient Descent Convergence
Before Scaling



Gradient Descent Convergence
After Scaling

- From the plots above, we can conclude that the convergence after scaling is more effective and finer than that of convergence before scaling.

CONCLUSION:

From all the above observations and results, I conclude that the Gradient Descent Function has successfully been implemented for learning a Logistic Regression model and was tested on unseen test data resulting in an accuracy of 87% when trained using 1 Million iterations for calculating the weights of the features.