**Gradient Descent for Logistic Regression**


- Name: B Sai Sahana Bhargavi
- Username on miner website : word2vec


**Introduction:** To implement Gradient Descent Algorithm for learning Logistic Regression model for the 152 records of Cleveland Dataset and classify the result of heart disease and compare the metrics with Sklearn's Logistic Regression model.

**Approach:**

1. **Data extraction:** Read the document containing 152 records of Cleveland Training Dataset and stored into a "Traindata_df" dataframe and 145 records of Testing Dataset and stored into a "Testdata_df".Each input record in Traindata_df is further stored into matrix X_train (training examples)and column vector Y_train (Labels), similarly,Testdata_df records into X_test.

2. **Gradient Descent Algorithm:**

   Defined following functions that are called in logistic regression using gradient descent algorithm. Here, x denotes independent variable matrix that represents all training data records, y - dependent variable matrix containing all classified labels (-1 or +1), m is total number of training samples.

   - **generate_vector(x):** contains all the independent training variables and adds one row that corresponds to x_0 (intercept to be computed)
   - **theta_init(x):** generates the initial values for the weight vector.
   - **sigmoid(z):** computes the sigmoid value for the input parameter
   - **logloss(y,X,theta,m):** computes cross entropy error.
   - **gradient(y,X,,theta,m):** computes gradient for logistic regression**.**
   - **Logistics_Regression(X,y,learningrate, iterations):**
     generates a Logistic Regression model. With the initial values of vector and theta returned by generate_vector(x) and theta_init(x) respectively, and further weight vector (theta) is updated for every iteration.,where learning rate = 0.00001, iterations=[ten thousand, one hundred thousand, and one million] and algorithm is repeated until it converges i.e if the magnitude of each term in the gradient reaches 0.001 at any step.

     **Formulas used for calculating:**

     Gradient:

     $$= -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i \mathbf{x}_i}{1+e^{y_i \mathbf{w}.\mathbf{x}_i}}$$

     Cross entropy error:

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

Weight vector(theta): theta get updated using the learning rate and the gradient calculated from the gradient function.

```
theta = theta - learningrate * gradients
```

Implemented model is run for 10000, 100000, 1000000 iterations and the corresponding results are recorded in the below table. Here, acccuracy for test data on miner obtained for 1000000 iterations is high (0.81) when compared to remaining iterations .

| Iteration(Train data) | 10000 | 100000 | 1000000 |
|---|---|---|---|
| Cross entropy error(Train data) | 0.9210 | 0.5432 | 0.407 |
| Classification(0/1) error(Train data) | 0.409 | 0.3114 | 0.229 |
| Accuracy on miner test data | 0.63 | 0.71 | 0.81 |
| Estimated classification error on the test data on the miner | 0.37 | 0.29 | 0.19 |
| Time taken to train model | 1 sec | 9 sec | 1min |

**3.Generalization properties:**

- On observing the training and test set classification errors for the model, it can be derived that the difference between the both classification errors is almost same and, also, the training and test sets classification errors reduces as the number of iterations increases. Obtaining, 0.229 error rate in final highest iteration bound 1000000.
- Similarly, cross entropy error also reduced as the number of iterations increased and minimizing cross entropy error ensures high accuracy of the training model and low classification errors.

**4. Logistic regression model using sklearn**

| Training data | Logistic regression using sklearn | 1000000(iterations) | 100000(iterations) |
|---|---|---|---|
| Cross entropy error | 0.3895 | 0.407 | 2.135 |
| Classification(0/1) error | 0.1622 | 0.229 | 0.344 |
| Estimated classification error on the test data on the miner | 0.84 | 0.81 | 0.71 |
| Time taken to train model | 0 sec | 1 min | 8 sec |

It can be observed that all the values obtained while using sklearn logistic regression model are similar to the ones recorded for the highest number of iterations 1000000.

**5.Feature scaling for the train data:**

Scaled the features of the training data by subtracting the mean and dividing by standard deviation for each feature before calling training function. I tried experimenting with various learning rates but the algorithm is continuously running without terminating for the given tolerance value of 10^-6. Then, i tried for different tolerance levels 0.5,0.01,0.001 with learning rates as 0.05,0.08,0.001,0.0001 still the gradient terms were not converged and algorithm is not terminating,unfortunately,due to above reasons, the results could not be recorded.

**Conclusion:** Based upon the results and comparisons, logistic regression with gradient decent model has more accuracy when there are higher number of iterations**, t**he model can be tuned further more with stochastic gradient descent where each gradient for each training sample is computed instead of computing gradient for whole training sample set and suggestable for large data set that ensures fast and less expensive approach where the convergence takes place faster.