Implement logistic regression to classify fruits images (Supervised Learning)

Keshav Choudhary
keshavc24@gmail.com
New Delhi, India
AI Technology and Systems, Intern
www.ai-techsystems.com

***Abstract*—Logistic Regression is a supervised machine learning algorithm use to classify data, it is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. In this project I collected Fruits dataset from Kaggle.com, the dataset name is Fruit360 it contains total 77,917 images in which 58,266 are train images and 19,548 test images, in which each image has a dimension of 100×100×3 and there are total 114 classes of fruits and vegetables. I performed logistic regression on this data and hyperparameter tuned it to get the best results.**

Keywords—Machine Learning, Logistic Regression, Image Classification.

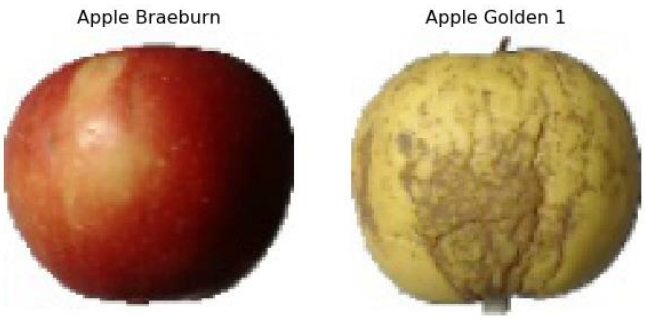
I. INTRODUCTION

Logistic Regression is a Supervised Machine Learning algorithm which is use to classify data, it is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. In this algorithm we train some weights which is equal to the no. of dimensions in data and a bias term, the main task is to reduce the loss which is multinomial log loss in our case and the algorithm through which it is done is called Gradient Descent. In this project I used logistic regression to classify fruits, I collected the dataset from Kaggle.com, this dataset contains total 77,917 images in which 58,266 are train images and 19,548 test images, in which each image has a dimension of 100×100×3 and there are total 114 classes of fruits and vegetables. I hyperparameter tuned my model on the training data and obtained the accuracy from test data which you can see in the code with this report and this I am going to explain everything about the model I created and the final results I got.

II. DATA PREPROCESSING

A. Data Collection

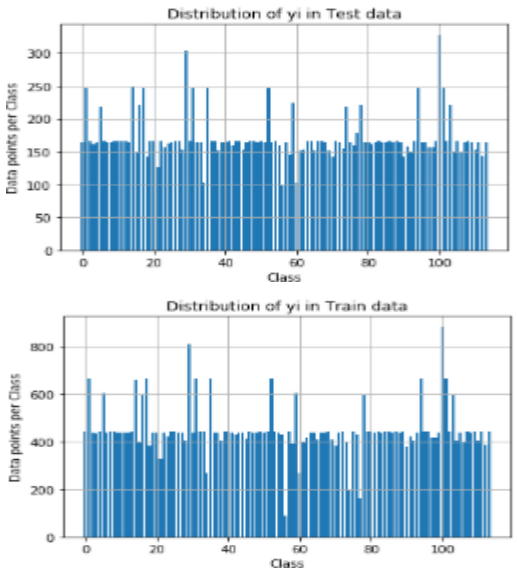
First, I downloaded the dataset from https://www.kaggle.com/moltean/fruits the dataset name is Fruit360. The folder contains train and test folders which contains 58,266 images and 19,548 images with 114 classes which are almost distributed equally, we can consider this dataset as balanced dataset. I collected the data in separate folders of images and labels where images are 30,000 dimensions vectors and labels as a unique integer per fruit. Some example of fruits is shown.



(Figure 1:- Image Sample**)**

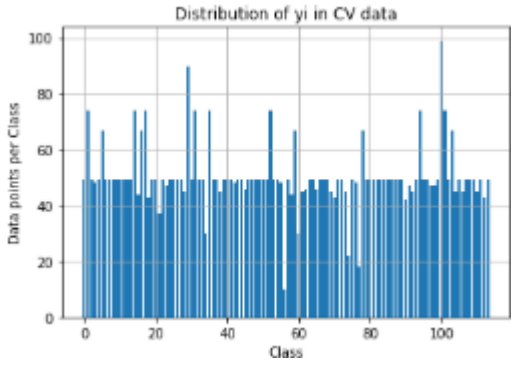
B. Train Test Split

After collection of all the data we split our dataset into Train, Test and Cross validation data we do this using sklearn library I took the test data as whole that we collected and then split the train data in the ratio 1:9 where training set has 51,548 data points and cross validation set has 5,728 data points. Then I plot the bar plot showing distribution of class labels in Train, Test and Cross validation data.



(Figure 2:-Test and Train label distribution**)**

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE



(Figure 3:- Cross validation labels distribution**)**

Here we can see that the distribution of classes in our Train, Test and Cross validation is almost equal and hence we can proceed further to train our models

III. MODEL

A. Logistic Regression

Logistic Regression is a Supervised Machine Learning algorithm which is use to classify data, it is a statistical method to analyze a dataset in which there are one or more independent variables that determine an outcome. In this algorithm we train some weights which is equal to the no. of dimensions in data and a bias term, the main task is to reduce the loss which is multinomial log loss in our case and the algorithm through which it is done is called Gradient Descent. Steps to follow to implement logistic regression.

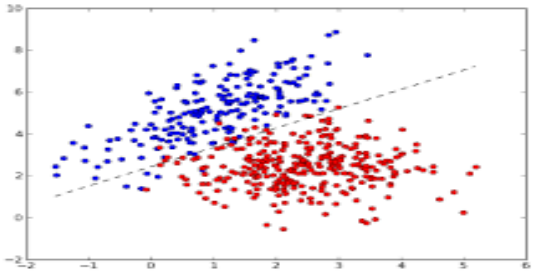
Step 1: - Initialize the weight vector to some random points in uniform distribution and assign 0 to bias term.

$$\pi = W^T X + B$$

The equation pi is called the hypothesis function which represents the plane which separates our classes it can separate only two classes. Here X denotes array of all features of data point which are 30,000 in our case and same is the dimension of weight vector and our bias term is an integer Our aim is to find out optimal values of weights which separate the classes.

Assumption: Classes are almost linearly separable.

Step 2: -



(Figure 4: - Line separating data)

Here, let the label of classes with blue color be -1 and the one with red be +1. Distance between an ith point and the plane is given by di.

$$d_i = W^T x_i + b$$

Classifier:

If: $d_i > 0, y_i = +1$

If: $d_i < 0, y_i = -1$

For correct classification of points we want

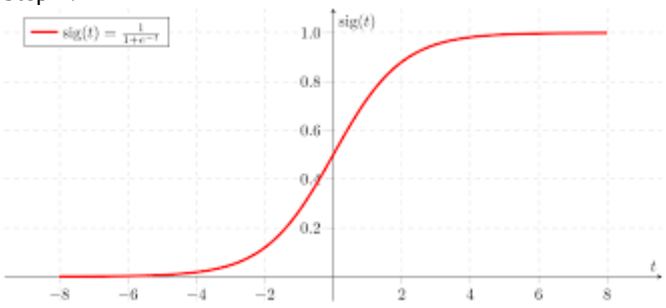
$$y_i(W^T x_i + b) > 0$$

Step 3: - Let W* be our optimized weight. So, our optimization problem is:

$$W^* = \arg\max_W \left(\sum_{i=1}^{n} y_i(W^T x_i + b)\right)$$

Here n is the number of points in Training data.

Step 4: -



(Figure 5:- Sigmoid Function)

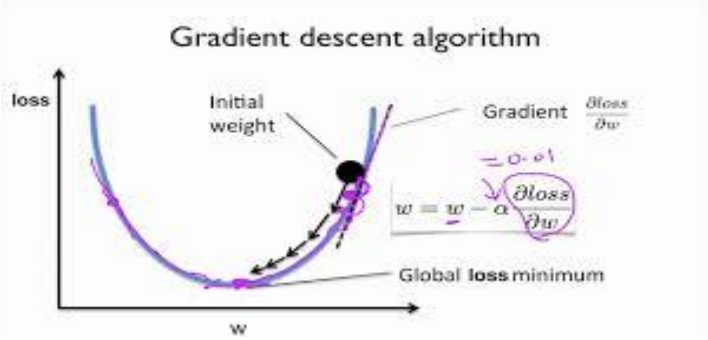
This is a sigmoid function represented as f(x).

$$f(x) = 1/(1+e^{-x})$$

The problem with the using of distance function directly is that it grows largely when encountered with outliers, so we want a function which should not increase further for large distances so we introduced SoftMax function and our new optimization problem becomes.

$$W^* = \arg\max_W \left(\sum_{i=1}^{n} (1/(1+e^{(-y_i(W^T x_i + b))}))\right)$$

Step 5: - Then we perform gradient descent algorithm to get the best weights it is as follows.



(Figure 6: - Gradient Descent)

Here alpha represents the learning rate which tells us how fast the convergence will take place and the loss function is same as the optimization function, but here we minimize it by just inversing the optimization function. The formula

is given as

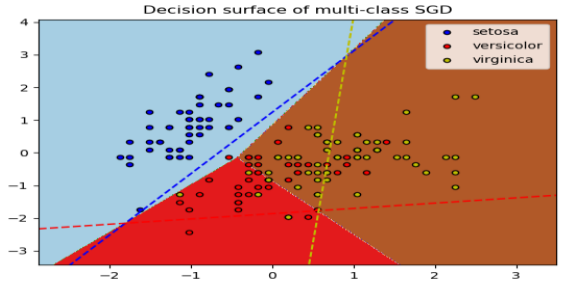
$$\text{Repeat until convergence } \{$$

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$\}$$

(Figure 7: - Gradient Descent Formula)

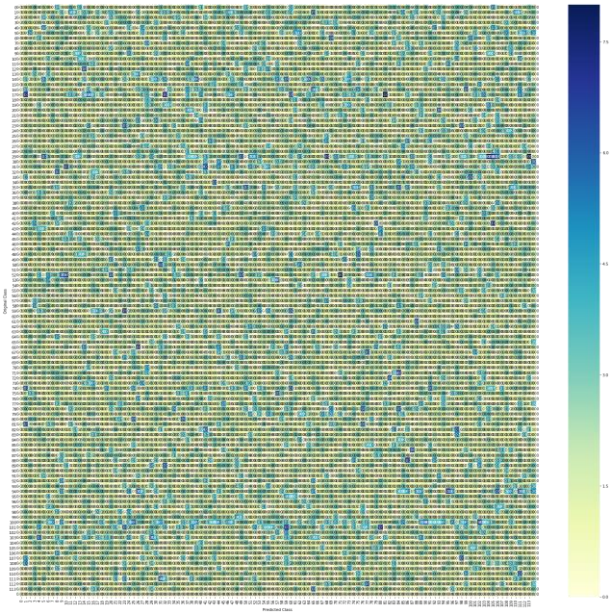
Step 6: - The algorithm discussed above is use to classify binary data, so for doing multiclass classification sklearn SGD classifier do one vs rest technique to find out correct label it works as. For each of the K classes, a binary classifier is learned that discriminates between that and all other K−1classes. At testing time, we compute the confidence score (i.e. the signed distances to the hyperplane) for each classifier and choose the class with the highest confidence. The Figure below illustrates the OVA approach on the iris dataset. The dashed lines represent the three OVA classifiers; the background colors show the decision surface induced by the three classifiers.



(Figure 8: - Decision surface for one vs rest)

B. Project Procedure

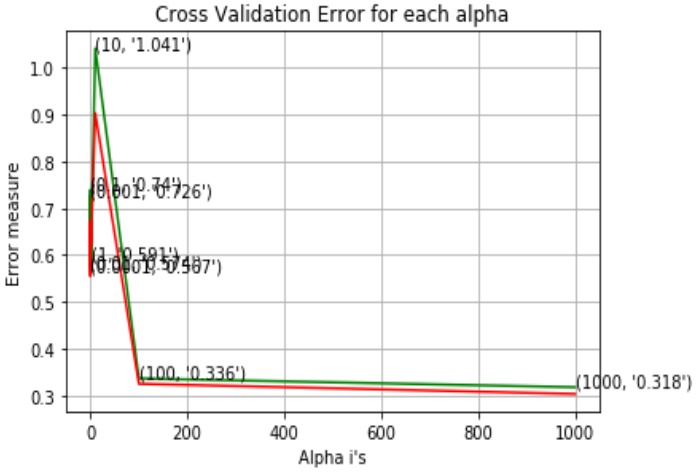
First, I find the log loss of the random model to get to know the cutoff as log loss lies in between zero and infinity and the loss obtained is 5.032 on cross validation data and 5.041 for test data and the confusion matrix we obtained is shown below.



(Figure 9: - Confusion Matrix for random model)

Here we can clearly see the randomness and how bad the model is so we get the cutoff to be 5.

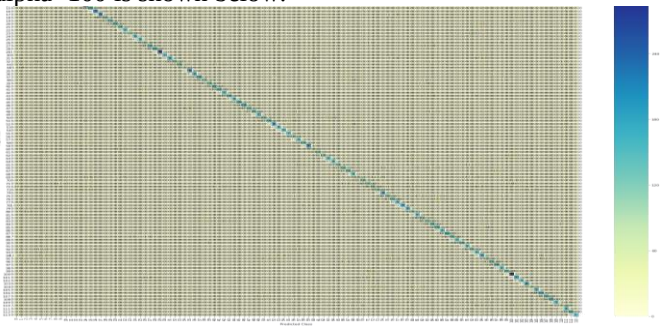
Now I trained my model using sklearn SGD classifier with L2 regularization and hyperparameter tuned it for learning rate values as 10**-4 to 10**3 in multiple of 10.



(Figure 10: - Loss vs alpha)

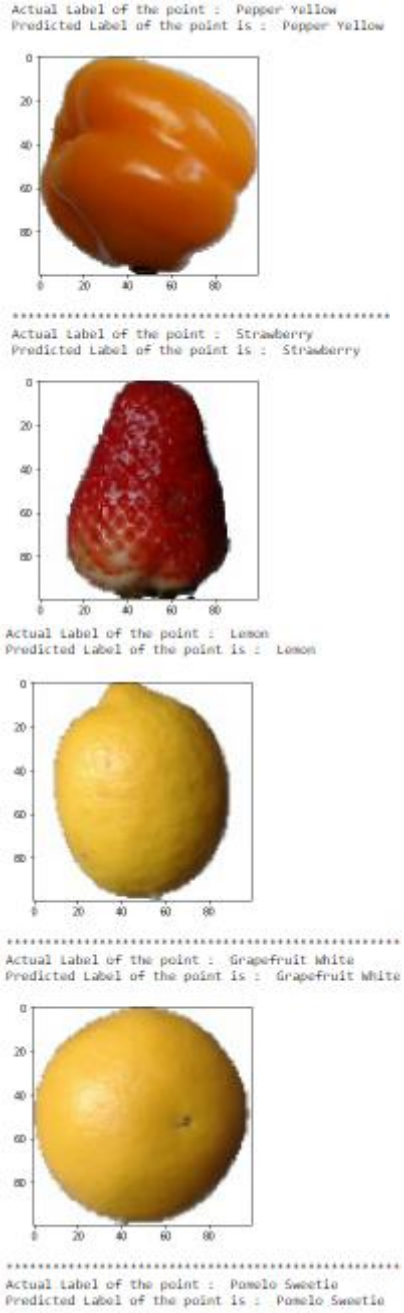
```
for alpha = 0.0001
Log Loss : 0.567439274427627
Accuracy : 0.9633379888268156
for alpha = 0.001
Log Loss : 0.725954016346483
Accuracy : 0.9495460893854749
for alpha = 0.01
Log Loss : 0.5738252144406923
Accuracy : 0.9636871508379888
for alpha = 0.1
Log Loss : 0.7395514132764188
Accuracy : 0.9455307262569832
for alpha = 1
Log Loss : 0.5907990046807491
Accuracy : 0.9514664804469274
for alpha = 10
Log Loss : 1.0413778691856912
Accuracy : 0.9167248603351955
for alpha = 100
Log Loss : 0.3364890796790490
Accuracy : 0.9472765363128491
for alpha = 1000
Log Loss : 0.3177831756000262
Accuracy : 0.9718924581005587
```

These are the values on Cross Validation data. So, I tried two hyper parameters on test data which are .0001 and 100. For alpha=.0001 I got test log loss as 2.52 and accuracy 84.27% and for alpha=100 I got test log loss as 0.79 and accuracy 86.8%. The confusion matrix for alpha=100 is shown below.



(Figure 11: - Confusion Matrix for alpha=100)

Here we can clearly see the dark diagonal line showing the correct predictions. Similarly, I trained SGD classifier with L1 regularization but got no better result.
Some examples of the prediction made by best model with alpha =100.



(Figure 12: - Sample Prediction)

IV. Conclusion

This paper shows the result of classification of fruit dataset using Logistic Regression where I hyperparameter tuned my model on both L1 and L2 regularizer. We got the best hyperparameter as 100 which is trained with L2 regularizer and for 10 iterations and we got the final highest accuracy as 86.8% and final log loss as 0.79.

.

V. References

1. https://www.kaggle.com/moltean/fruits
2. https://scikit-learn.org/stable/modules/sgd.html