Logistic Regression to classify Fruits Image Dataset

Implementing Logistic Regression with Fruits dataset at https://www.kaggle.com/moltean/fruits.

Ganesh Kumar T K
Machine Learning Engineer Intern.
AI Tech Systems.
www.ai-techsystems.com
Chennai, India.
Kichulee1998@gmail.com

***Abstract* — Recent developments in artificial intelligence technology have shown good results in various fields. This paper aims to construct and implement Logistic Regression for classifying fruits based on Supervised Classification. Here, I used open source library interface "Scikit-learn" to classify a single fruit grade, and application of logistic regression gives a training accuracy and testing accuracy of 100%. Using this system in the future, consumers will be able to choose quality fruits that are price appropriate.**

***Keywords* — Unsupervised learning, Logistic regression, Fruits, Image Classification**

I. INTRODUCTION

According to Arthur Samuel, Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed. The machine learns by analyzing the given data and obtains a decision boundary and classifier for solving classification and prediction problems. As such, machine learning technique finds application in numerous fields and areas of study for data analysis as it can successfully classify and predict some unknown event based on known data. Image Classification is one important field of study where certain methods of machine learning are employed given the captured image or the image feature as an input

Regression analysis is the process of estimating the relationships among variables and predicting where a particular variable belongs to which class. Regression is basically a two-class classification method. It is broadly of two types: Linear and non-linear regression. Linear regression analysis has a classifier which is represented by a straight line while non-linear regression analysis has a classifier represented by a curve. Logistic regression allows non-linear boundary model for a two-category classification unlike linear regression which works only when two classes can be separated by a linear line. Linear regression analysis is used for face recognition system successfully if the two class of samples can be linearly separated. It is also employed with equable principal component analysis for image recognition; however, logistic regression works even when a two class of samples cannot be linearly separated thus generating a chance of higher rate of correct classification. Logistic regression uses a logistic function (also known as a sigmoid function) which produces a “S”-shaped curve in the range between 0 and 1, making it possible to obtain non-linear boundary as shown in figure.

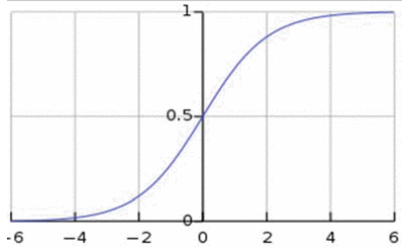


Fig.1 Logistic curve or sigmoid curve

In logistic regression, a hypotheses function $h_\theta(x)$ is used as a threshold classifier which predicts y=1when $h_\theta(x) \geq 0.5$ and y=0 when $h_\theta(x) < 0.5$. Hypotheses function classifies the input features into two classes [0 and 1]. The matrix X is transposed ($X=X^T$) so that feature matrix is in column vector.

$$X^T = \begin{bmatrix} 1 & 1 & & 1 \\ x_j^1 & x_j^2 & \cdots & x_j^m \end{bmatrix} \text{ where j} = 1 \text{ to n}$$

II. DATA PRE-PROCESSING

The Fruit-360 dataset from kaggle.com consist of 59,328 images of fruits in the Training folder. These images are divided into 118 classes of fruits and each class contains approximately 400 images. Shape of every image is 100x100x3 pixels with 3 colour channels (RGB).

Fig.2 Fruit-360 Data-set

A. Binarization and Fruit Region Detection along with feature extraction:

Here, color image is first converted into grayscale image which is then converted to binary image. Getting a high-quality binary image is very crucial to obtain the prominent feature of the image.

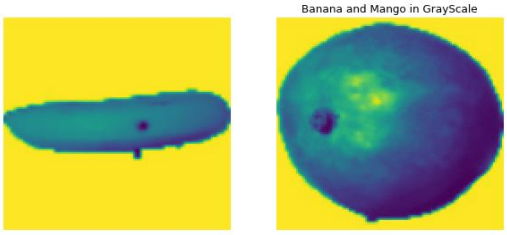
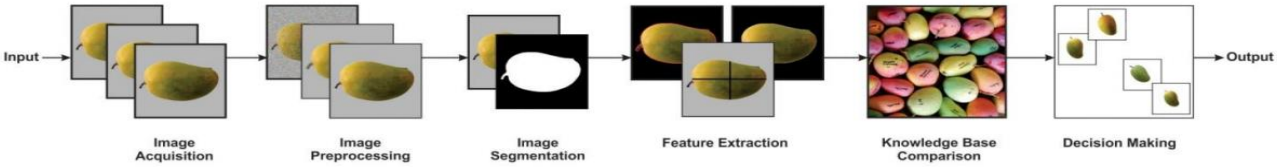


Fig.3 Grayscaling of RGB image



Here, we first convert color to gray scale image. We obtain a binary image by applying local window standard deviation (LWSD) and adaptive thresholding to gray scale image. Local window standard deviation for an image X using a sliding window of size MxN can be computed as

$$\Sigma = \sqrt{\frac{1}{MN-1}(q - \frac{s^2}{MN})}$$

B. Feature Extraction, Resizing and Vectorization of Image

Feature extraction is an important component of face recognition system as the success rate is highly dependent to how well the feature extracted represents the image uniqueness to different class and its likeness to images of the same class. One such feature extraction method is to employ local binary pattern for image recognition. In holistic approach, image of the whole fruit is used instead of local features. So we have to input all the image pixels as a feature vector for learning. The binary face image extracted here has a size of 100 × 100 = 10000 pixels which is too large and not feasible for training. To reduce the complexity and reduce computational time of learning, we apply nearest neighbor interpolation.

The reduced image obtained by nearest neighbor interpolation has a size of m by n where m and n are the number of rows and columns of the reduced image matrix. The reduced image matrix is converted into a row vector by concatenating each row one after another horizontally. A feature data set X is formed by stacking each row vector of the images as,

$$X = \begin{bmatrix} image\ 1\ \ vector \\ image\ 2\ \ vector \\ \cdots \\ image\ m\ \ vector \end{bmatrix}$$

A column vector Y of m×1 dimension is formed with labels (target value) given for each corresponding image vector in X where m is the number of sample size. Same label number is given to images of the same person so as to have image of one person as one class and image of the other person as another class, and so on. In machine learning, the matrix X will be input feature and the vector y is the output label for each row.

```
def train_data():
    train_data_b = []
    train_data_m=[]
    for image1 in tqdm(os.listdir(train_b)):
        path = os.path.join(train_b, image)
        img1 = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
        img1 = cv2.resize(img1, (image_size, image_size))
        train_data_b.append(img1)
    for image2 in tqdm(os.listdir(train_m)):
        path = os.path.join(train_m, image)
        img2 = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
        img2 = cv2.resize(img2, (image_size, image_size))
        train_data_m.append(img2)

    train_data= np.concatenate((np.asarray(train_data_b),np.asarray(train_data_m)),axis=0)
    return train_data
```

Fig.4 Resizing and Feature scaling of image

III. LOGISTIC REGRESSION ALGORITHM

Logistic regression uses a logistic function (also known as a sigmoid function) which produces a "S"-shaped curve in the range between 0 and 1, making it possible to obtain non-linear boundary.

The sigmoid function g(z) where z is some parameter values can be given as

$$g(z) = \frac{1}{1+e^{-z}}$$

The learning parameter θ is a column vector of (n+1) by1 and initialized to 0' s. The learning parameter θ will change its value in each iteration to fit the model. Now, the hypotheses function can be calculated as

$$h_\theta(x) = g(\theta^T x)$$
$$\text{such that } 0 \leq h_\theta(x) \geq 1$$

where x is the vector elements of matrix X and θ the learning parameter value associated with each pixel value in the feature matrix. In machine learning, a cost function measure how far a particular solution is away from optimum solution and a gradient descent is the process of taking steps proportional to the negative of gradient towards an optimum solution (or local minima). A cost function and a gradient descent may be iterated a number of times to learn from the given data and to calculate the hypotheses function.

To get an optimum solution to generate a non-linear boundary, the regularized cost function is iterated number of times until the gradient descent falls to the local minima. The decision boundary generated for two-class classification is shown in figure.

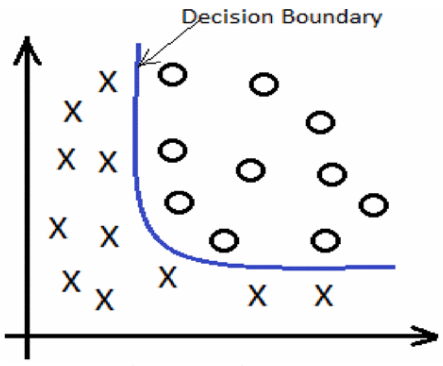


Fig5. Classified output of Logistic Regression

Applying the given formula, we can train the machine to classify one class from others (one-vs-all) separately. Therefore, for a given k number of classes, we have to train *n* number of classifier, one for each class. An iterative method is used to calculate the cost function and gradient descent for a specified number of times or till it converge to optimum solution. The training of data produces an output y_{out}={0 or 1} which is called prediction.

A. *Proposed Algorithm*

The steps involved in the proposed method are depicted in the algorithm given below.

Algorithm:

1. Load RGB image.
2. Convert RGB image to Greyscale image.
3. Taking the Hue component image and subtract its background.
4. Obtain 'T' threshold using OTSU's method.
5. Extract the ROI using the same and fill the holes.
6. Image with ROI alone is loaded.
7. Features like statistical and textural features were extracted using Wavelet Transformation using Scikit-learn library for all Vertical, horizontal and diagonal coefficient values.
8. Extracted features from both training and testing samples were given as input to Logistic regression classification procedure.
9. Fruits images were classified.

IV. RESULTS

In this paper, we use Fruit-360 image database which is freely distributed for research. Here, we consider only the frontal image and slightly tilted image for training and testing purposes. Therefore, we have taken 490 images for training and 196 images for testing in this experiment. Logistic Regression values of training and its corresponding testing dataset are given in figure:

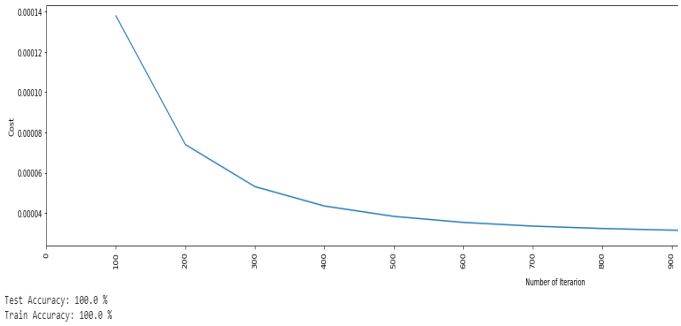


Fig.6 Sigmoid Curve obtained for given test data.

This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e., not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomised starting centroids may give a better outcome.

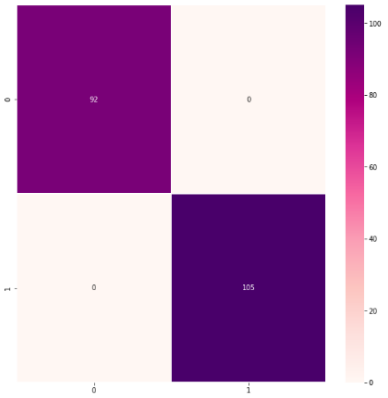


Fig.7 Confusion Matrix Obtained with values 92 and 105.

V. CONCLUSION AND FUTURE SCOPES

One of the clusters formed is shown below: Images from Fruits-360 dataset alone used for testing this algorithm using Logistic Regression classifier.
Sample images should be acquired at 360 degrees in order to obtain 100% accuracy in real time classification of any fruit in the agriculture industry.

Thus, Logistic Regression for segregating/ classifying fruits was developed and tested for 100% accuracy and the same was obtained successfully.

Sl No	Size of image	Percentage of size reduction from original size	**Logistic Regression**	
			Training Accuracy	**Testing Accuracy**
1	95X85	50%	100%	100%
2	57X51	30%	100%	100%
3	38X34	20%	100%	100%
4	19X17	10%	100%	97.5%

Fig.8 Logistic Regression for Various Image Size

Also, this research work can be extended to help the agriculturist to classify different varieties of fruits.

VI. REFERENCES

1. https://www.kaggle.com/moltean/fruits
2. https://www.geeksforgeeks.org/clustering-in-machine-learning/
3. https://www.datascience.com/blog/logistic-regression
4. https://fast.ai/en/popular-networks/vgg16/
5. https://towardsdatascience.com/a-step-by-step-approach-to-logistic-regression--b836fb9c97e2