| **Ex No: 1**  **Date:** | **Multi-Class Classification using Softmax Regression and Gradient Descent** |
| --- | --- |

# 1. Objective:

The primary objective of this exercise is to build a multi-class classification model to recognize and classify different types of flowers using a softmax regression approach implemented with gradient descent. The aim is to accurately categorize images of flowers into one of several predefined classes by optimizing the model parameters through iterative training.

# 2. Description:

In this multi-class classification problem, we are tasked with recognizing and categorizing images of various flowers into one of five distinct classes: ‘Rose’, ‘Tulip’, ‘Sunflower’, ‘Daisy’, and ‘Dandelion’.Here, we need to classify images into multiple classes, each representing a different type of flower.

## 2.1 Multi Class classification:

Multi-Class Classification is a type of classification problem in machine learning where the task is to categorize instances into one of three or more predefined classes. Unlike binary classification, which involves only two possible outcomes (e.g., spam vs. not spam), multi-class classification deals with scenarios where there are multiple possible categories or labels.

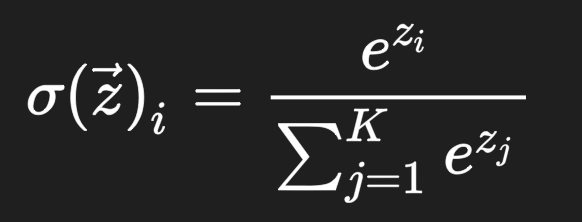
### 2.1.1 Classes/Labels:

In a multi-class classification problem, the target variable has more than two categories. For example, in this flower classification problem, the classes could be ‘Rose’, ‘Tulip’, ‘Sunflower’, ‘Daisy’, and ‘Dandelion’.

### 2.1.2 Softmax Regression:

Softmax regression, also known as multinomial logistic regression, is an extension of binary logistic regression that handles multiple classes. It transforms raw output scores (logits) from the model into probabilities, ensuring that the sum of probabilities across all classes equals 1.

The softmax function is defined as:

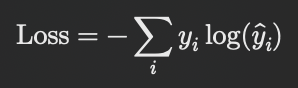


where z\_i is the raw score for class i , and e is the base of the natural logarithm. The function exponentiates the scores, normalizes them, and produces a probability distribution over all possible classes.

### 2.1.3 Cross-Entropy Loss Function

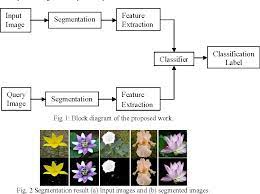
The cross-entropy loss function is used to measure the difference between the predicted probabilities and the actual class labels. It is a common loss function for multi-class classification problems and helps to quantify the model’s performance.

The cross-entropy loss is given by:



where y\_i is the true label (one-hot encoded), and y^\_i is the predicted probability for class i . The goal during training is to minimize this loss function to ensure that the predicted probabilities closely match the actual class labels.

# 3. Model

****

Multi Class Classification Model

## 3.1 Model Building Steps

### 3.1.1 Input Layer

For this multi-class flower classification problem, the input layer handles features extracted from the image data by flattening the 2D image into a 1D vector. Each image is 64x64 pixels with 3 color channels (RGB), resulting in an input layer with 64\*64\*3 = 12,288 neurons. The ImageDataGenerator is used for image preprocessing and augmentation before the images are flattened into a 1D vector for input.

### 3.1.2 Output Layer

The output layer is a dense layer with 5 units, corresponding to each flower class, and uses a softmax activation function. This layer produces a probability distribution across the five flower classes, where each unit represents one class. The softmax function is applied to convert raw scores into probabilities.

### 

### 3.1.3 Initialize Model Parameters:

The model parameters, including weights and biases, are initialized with small random values when defining each layer. This initialization is crucial to prevent symmetry and help the model converge faster.

## 3.2 Training Procedure

### 3.2.1 **Forward Propagation:**

During forward propagation, the input data is passed through the network to compute predicted probabilities. The softmax function is applied to the output layer to obtain a probability distribution over the five classes.

### 3.2.2 Compute Loss:

The categorical cross-entropy loss is calculated between the predicted probabilities and the actual class labels. This loss function measures the difference between the predicted and true distributions, guiding the model’s learning process.

### 3.2.3 Backward Propagation

Gradients of the loss function with respect to the model parameters are computed using backpropagation. This step helps determine how each parameter should be adjusted to reduce the loss.

### 3.2.4 Parameter Update

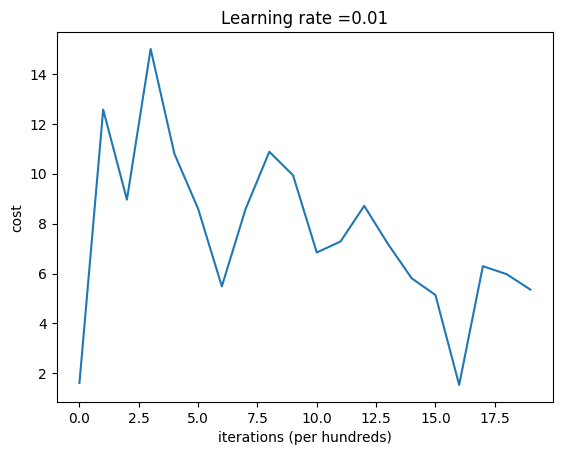
Gradient descent is employed to update the model parameters based on the computed gradients. The learning rate determines the step size for each update, influencing the speed and stability of convergence. The learning rate is set to 0.01 in the optimizer.

## 4. Evaluation

The model’s performance is evaluated using a separate test set that was not seen during training. Metrics such like accuracy is computed to assess the model’s effectiveness in classifying images into the correct flower categories. Visualization of learning curves and cost functions provides insights into the training dynamics and model behavior.

Train accuracy = 57.34958111195735 %

Test accuracy = 37.5 %



**GitHub Link:** [**https://github.com/Kashishvarmaa/DL-CS3232/tree/main/Lab\_1**](https://github.com/Kashishvarmaa/DL-CS3232/tree/main/Lab_1)