


# Data Science/Machine Learning/Deep Learning Workshop

*Nueva Ecija University of Science and Technology  
(NEUST)*

Rodolfo C. Raga Jr., PhD CS

*November 21-23, 2021*

A solid orange shape that starts as a thin wedge at the bottom left and expands diagonally upwards to the right, filling the bottom right corner of the slide.

# Day 2 : Neural Networks and Practical Application

## Topic 1: Fundamentals of Neural Network-based Simple Linear Regression (SLR)

- Introduction to Simple Linear Regression
- The simple linear regression model
- Implementing SLR using Neural Networks

## Topic 2: Fundamentals of Neural Network-based Multiple Linear Regression (MLR)

- Introduction to Multiple Linear Regression
- The multiple linear regression model
- Using categorical variables in regression models
- Implementing SLR using Neural Networks

## Topic 3: Fundamentals of Neural Network-based Binary Logistic Regression (BinLogRes)

- Introduction to Binary Logistic Regression
- The logistic regression model
- Implementing Binary LogRes using Neural Networks
- Performance metrics for logistic regression

## Topic 4: Fundamentals of Neural Network-based Multinomial Logistic Regression (MulLogRes)

- Introduction to Multinomial Logistic Regression
- The multinomial logistic regression model
- Assumptions of multinomial logistic regression
- Classifying Image Data using NN-based MLR implementation

# Introduction to Multinomial Logistic Regression

- ◆ Multinomial logistic regression is used to predict categorical placement in or the probability of category membership on a dependent variable based on multiple independent variables.
- ◆ The independent variables can be either dichotomous (i.e., binary) or continuous (i.e., interval or ratio in scale).
- ◆ Multinomial logistic regression is a simple extension of binary logistic regression that allows for more than two categories of the dependent or outcome variable.
- ◆ Like binary logistic regression, multinomial logistic regression uses maximum likelihood estimation to evaluate the probability of categorical membership.

# Multinomial Logistic Regression by any names

Multinomial logistic regression is known by a variety of other names:

- ➡ Conditional maximum entropy model,
- ➡ Maximum entropy classifier,
- ➡ Multiclass logistic regression.
- ➡ Multinomial logit,
- ➡ Polytomous logistic regression,
- ➡ Softmax regression.

# Characteristics of Multinomial Logistic Regression

- ❖ The dependent variable has more than two possible outcomes
- ❖ The dependent variable is nominal: there is no order in the outcome.
- ❖ The independent variables are used to predict the outcome.

# Examples of Binary vs. Multi Classification

## ❖ **Binary Classification:**

- ➡ Given the subject and the email text predicting, Email Spam or not.
- ➡ Sunny or rainy day prediction, using the weather information.
- ➡ Based on the bank customer history, Predicting whether to give the loan or not.

## ❖ **Multi-Classification:**

- ➡ Given the dimensional information of the object, Identifying the shape of the object.
- ➡ Identifying the different kinds of vehicles.
- ➡ Based on the color intensities, Predicting the color type.

# Sigmoid vs Softmax function

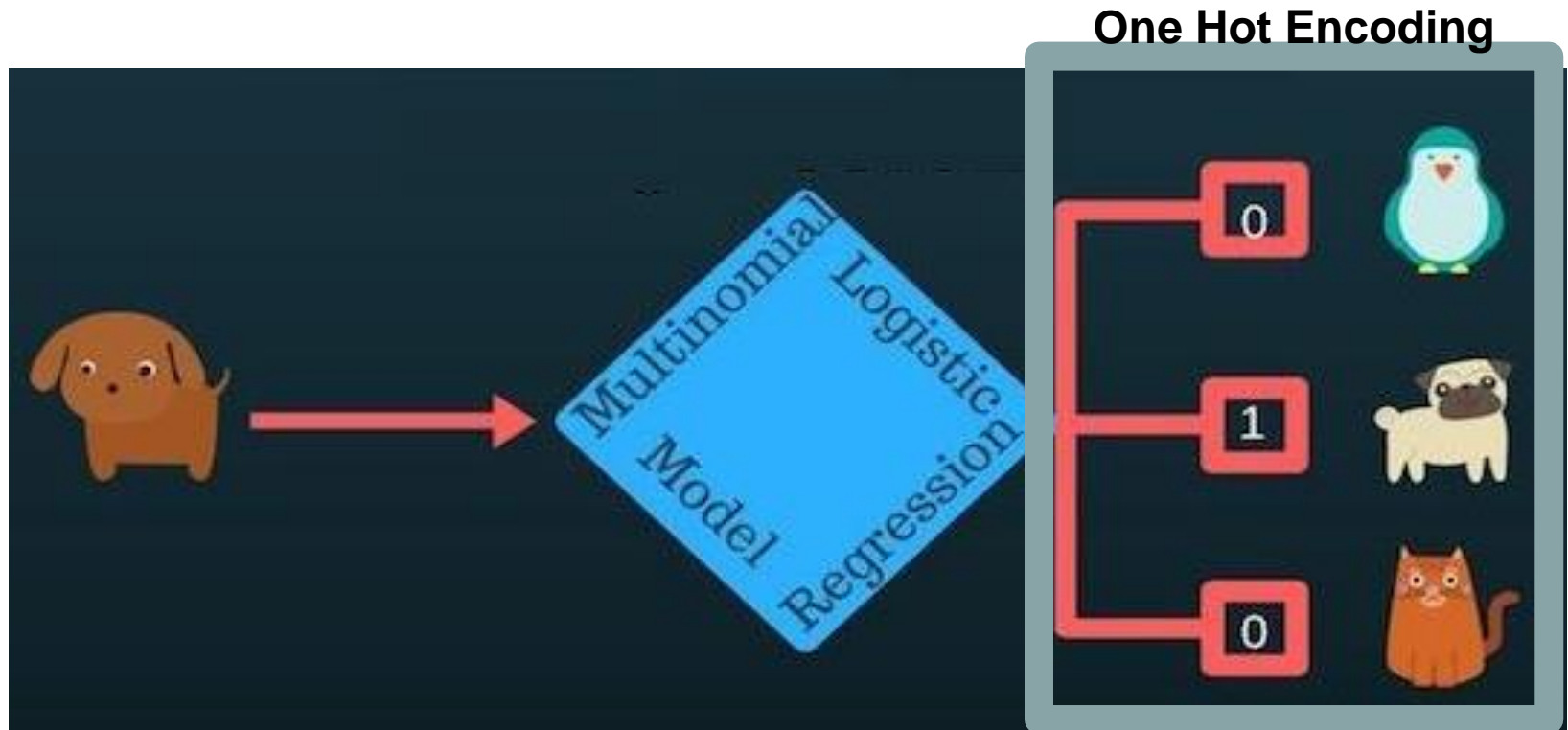
- ❖ In the logistic regression, the black function which takes the input features and calculates the probabilities of the possible two outcomes is the **Sigmoid Function**. Later the high probabilities target class is the final predicted class from the logistic regression classifier.
- ❖ When it comes to the multinomial logistic regression the function is the **Softmax Function**.

# The Softmax function

- ❖ **Softmax function** calculates the probabilities distribution of the event over  $n$  different events. In a general way, this function will calculate the probabilities of each target class over all possible target classes.
- ❖ The sum of all the probabilities will be **equal to one**
- ❖ The calculated probabilities also serves as basis for determining the target class for the given inputs.



# The Softmax Function



# One Hot Encoding

- ◆ One-Hot Encoding is a method to represent the target values or categorical attributes into a binary representation. From this article main image, where the input is the dog image, the target having 3 possible outcomes like **bird, dog, cat**. Where you can find the one-hot-encoding matrix like  $[0, 1, 0]$ .
- ◆ The one-hot-encoding matrix is so simple to create. For every input features ( $x_1, x_2, x_3$ ) the one-hot-encoding matrix is with the values of 0 and the 1 for the target class. The total number of values in the one-hot-encoding matrix and the unique target classes are the same.
- ◆ Suppose if we have 3 input features like  $x_1, x_2$ , and  $x_3$  and one target variable (With 3 target classes). Then the one-hot-encoding matrix will have 3 values. Out of **3** values, one value will be **1** and all other will be **0s**.
- ◆ You will know where to place the 1 and where to place the 0 value from the training dataset. Let's take one observation from the training dataset which contains values for  $x_1, x_2, x_3$  and what will be the target class for that observation. The one-hot-encoding matrix will be having 1 for the target class for that observation and 0s for other.

# About Image Classification

To enable computers to interpret images in the way that humans do, we have to somehow convert the images to numbers for the computer to understand.

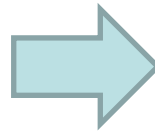
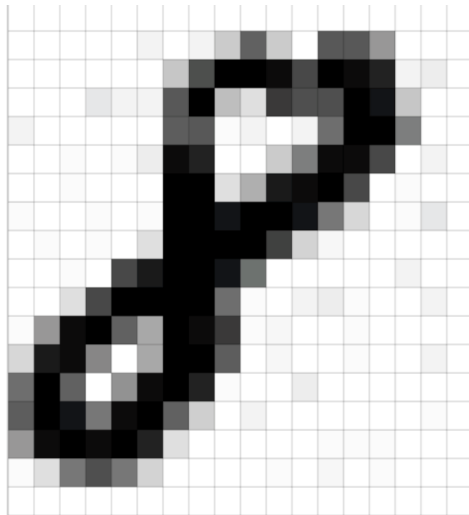
There are two common ways to do this in Image Processing:

1. Using Greyscale
2. Using RGB Values:

# Using Grayscale

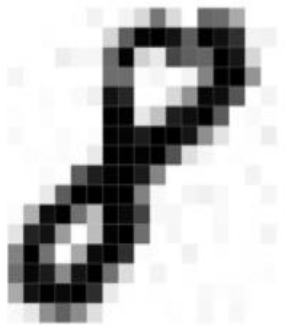
Using the greyscale technique, every pixel of an image will be converted to a range of gray shades and the computer will assign a corresponding numeric value to each pixel based on how dark it is. All the numbers are then put into an array and the computer does computations on that array.

For example, this is how the number 8 is seen by the computer using the Grayscale technique:

[illegible]

# Using Grayscale

The grayscale matrix is then flattened into one a single vector (array) structure. The resulting vector, as shown below, can then be fed into the computer as input:



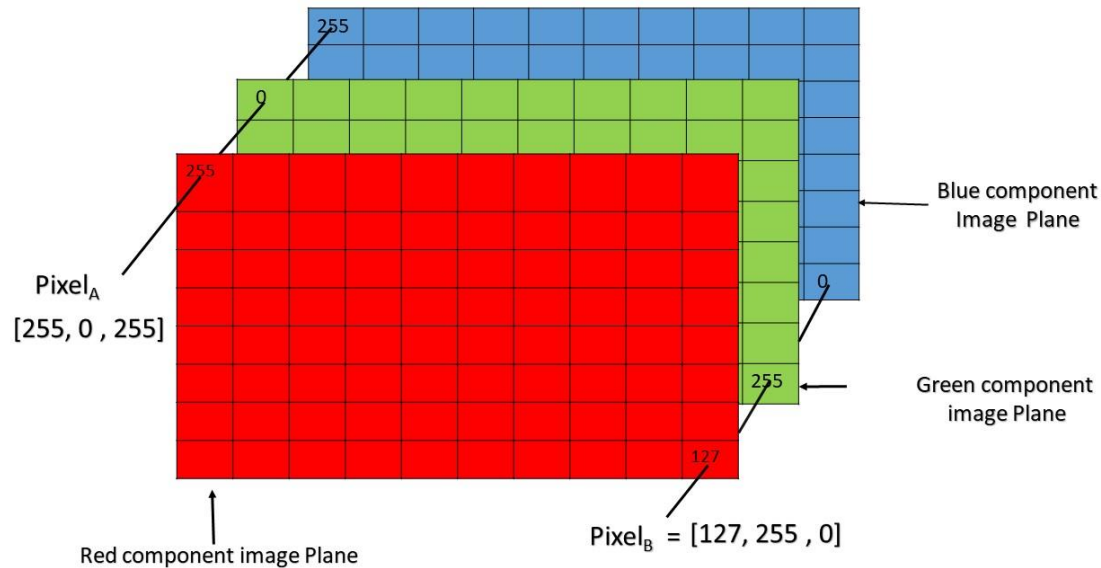
=

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 12, 0, 11, 39, 137, 37, 0, 152, 147, 84, 0, 0, 0, 0,
0, 1, 0, 0, 0, 41, 160, 250, 255, 235, 162, 255, 238, 206, 11, 13, 0, 0, 0, 0, 16, 9, 9, 150, 251, 45, 21, 184, 159, 154, 2
55, 233, 40, 0, 0, 10, 0, 0, 0, 0, 0, 0, 145, 146, 3, 10, 0, 11, 124, 253, 255, 107, 0, 0, 0, 0, 3, 0, 4, 15, 236, 216, 0, 0,
38, 109, 247, 240, 169, 0, 11, 0, 1, 0, 2, 0, 0, 0, 253, 253, 23, 62, 224, 241, 255, 164, 0, 5, 0, 0, 6, 0, 0, 4, 0, 3, 252
, 250, 228, 255, 255, 234, 112, 28, 0, 2, 17, 0, 0, 2, 1, 4, 0, 21, 255, 253, 251, 255, 172, 31, 8, 0, 1, 0, 0, 0, 0, 0, 4,
0, 163, 225, 251, 255, 229, 120, 0, 0, 0, 0, 0, 11, 0, 0, 0, 0, 21, 162, 255, 255, 254, 255, 126, 6, 0, 10, 14, 6, 0, 0, 9
, 0, 3, 79, 242, 255, 141, 66, 255, 245, 189, 7, 8, 0, 0, 5, 0, 0, 0, 0, 26, 221, 237, 98, 0, 67, 251, 255, 144, 0, 8, 0, 0
, 7, 0, 0, 11, 0, 125, 255, 141, 0, 87, 244, 255, 208, 3, 0, 0, 13, 0, 1, 0, 1, 0, 0, 145, 248, 228, 116, 235, 255, 141, 34
, 0, 11, 0, 1, 0, 0, 0, 1, 3, 0, 85, 237, 253, 246, 255, 210, 21, 1, 0, 1, 0, 0, 6, 2, 4, 0, 0, 0, 6, 23, 112, 157, 114, 32
, 0, 0, 0, 0, 2, 0, 8, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Resulting Array (src)

# Using RGB values

Images can also be represented as RGB values (a combination of red, green and blue ranging from 0 to 255). The same as with the Grayscale technique, computers can then extract the RGB value of each pixel and place the result in a vector for interpretation.



Pixel of an RGB image are formed from the corresponding pixel of the three component images

# Image Processing using a Neural Network

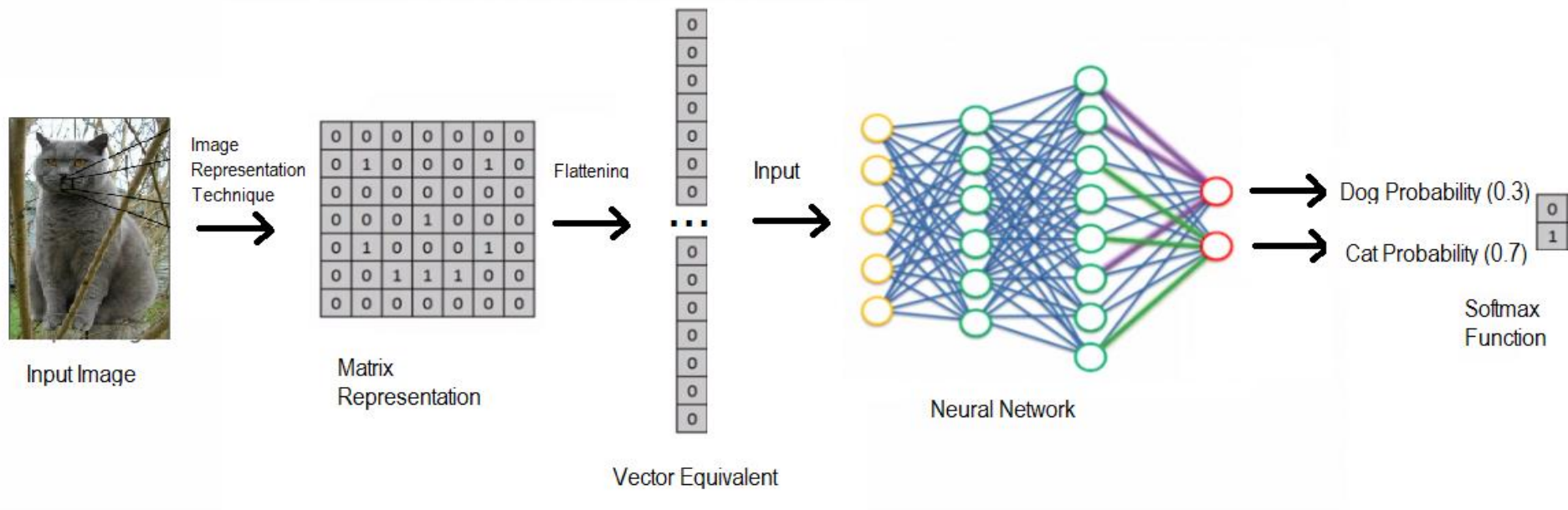


Image Processing Pipeline using a Neural Network

# Neural Network for Multinomial Regression

# Random Demo Time !!!