Universidad Nacional de Ingenieria

National Peruvian University

Prediction of Numbers

by Logistic Regression and Networks Neuronal

Instructor: Pacheco Taboada Andre

November 2, 2023

Abstract

Contents

1 Introduction

The task of digit recognition, which is a subset of Optical Character Recognition (OCR), has been a significant topic of interest within the Machine Learning community. The ability to accurately identify handwritten or printed digits from images can greatly contribute to various automation systems and is essential in fields such as banking, postal services, and document management systems. This project aims to develop a robust digit recognition system through the application of Multinomial Logistic Regression and Neural Networks, capable of processing images containing digits and predicting the respective numeric values with high accuracy.

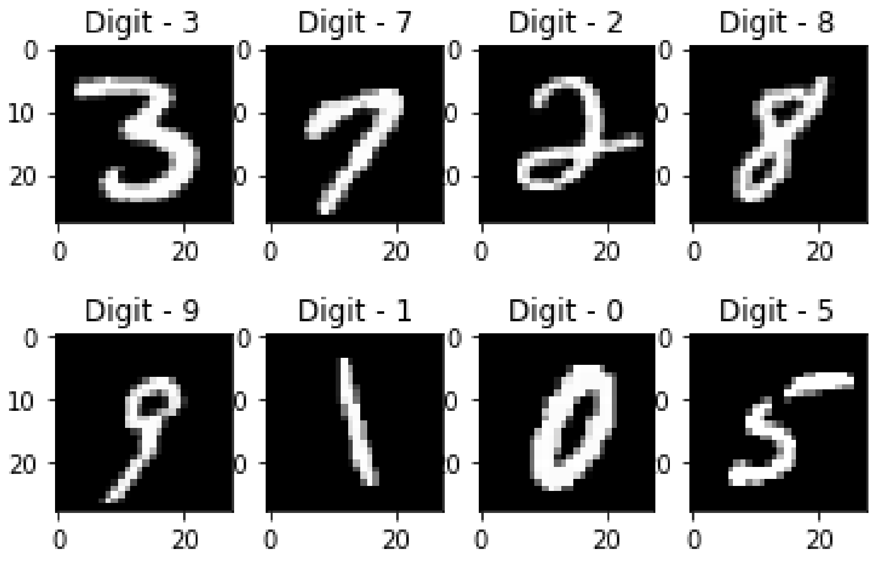

Figure 1: Plot of a Subset of Images From the MNIST Dataset

To achieve this, the project employs Flask, a lightweight WSGI web application framework, for developing a user-friendly interface where users can upload images of digits or manually input digits for prediction. Python, a versatile and powerful programming language, is used for developing the Machine Learning models and handling data processing tasks. The following sections will delve into the theoretical basis and methodologies adopted in this project, focusing on the use of Multinomial Logistic Regression and Neural Networks to meet the stated objectives.

2 Objectives

The primary objectives of this project are outlined as follows:

1. To develop a Multinomial Logistic Regression model capable of predicting digits from images with a high degree of accuracy.

2. To design and train a Neural Network to further enhance prediction accuracy and compare its performance with the Multinomial Logistic Regression model.

3. To implement a user-friendly web interface using Flask, enabling users to upload digit images or input digits manually for prediction.

4. To rigorously evaluate the performance of the developed models, ensuring they meet the required standards for practical deployment.

3 Theoretical Framework

The theoretical foundation of this project lies in the principles of Machine Learning, particularly in Logistic Regression, Neural Networks, and more specifically, Convolutional Neural Networks (CNNs). Understanding these algorithms is crucial for developing a predictive model capable of recognizing digits from images.

3.1 Machine Learning Fundamentals

Machine Learning, a subset of Artificial Intelligence, primarily focuses on developing algorithms capable of learning from data to make predictions or decisions. This field encompasses various techniques, ranging from simple linear regression to complex deep learning models.[1]

3.1.1 Supervised Learning

Supervised Learning involves training a model on labeled data, where the algorithm learns patterns and relationships between input and output. Common supervised learning methods include regression and classification algorithms. Multinomial Logistic Regression is an example of a supervised learning technique used when classifying data into multiple categories.

3.1.2 Unsupervised Learning

In contrast, Unsupervised Learning works with unlabeled data to discover hidden patterns or structures. Clustering and dimensionality reduction are typical unsupervised learning methods.

Machine Learning encompasses various categories, including but not limited to Supervised and Unsupervised Learning. Understanding these concepts is crucial, as Supervised Learning techniques like Multinomial Logistic Regression will form a cornerstone of the upcoming project.[1] [1]

3.2 Logistic Regression

Logistic Regression is a fundamental statistical technique used for binary classification problems. It's a method employed when the outcome variable is categorical, typically representing two classes. The logistic regression model estimates the probability that a given input belongs to a particular class.

In Logistic Regression, the relationship between the dependent variable and one or more independent variables is modeled using the logistic function, also known as the sigmoid function. This function maps any real-valued number to a value between 0 and 1, representing a probability.

The formula for the logistic regression model is:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Where: - $P(Y = 1|X)$ is the probability of the dependent variable Y being class 1 given the input X. - β_0 is the intercept term. - β_1 is the coefficient for the input variable X. - e is the base of the natural logarithm.

3.3 Multinomial Logistic Regression

Multinomial Logistic Regression, an extension of logistic regression, is used for multi-class classification problems. Unlike binary logistic regression, it can handle scenarios where the outcome can belong to three or more categories.[2]

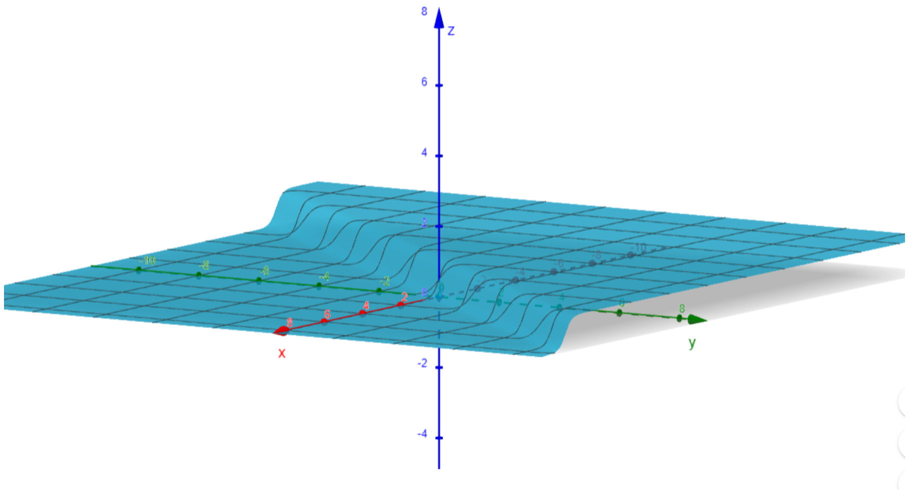

Figure 2: Graphical Representation of Multinomial Logistic Regression

3.4 Neural Networks

Neural Networks are computational models inspired by the structure and function of the human brain. They consist of layers of interconnected nodes or neurons and excel at learning intricate patterns and relationships in data.[3]

3.4.1 Architecture

A basic neural network consists of interconnected layers: input, hidden, and output. The input layer receives data; the hidden layers process information; and the output layer produces predictions.

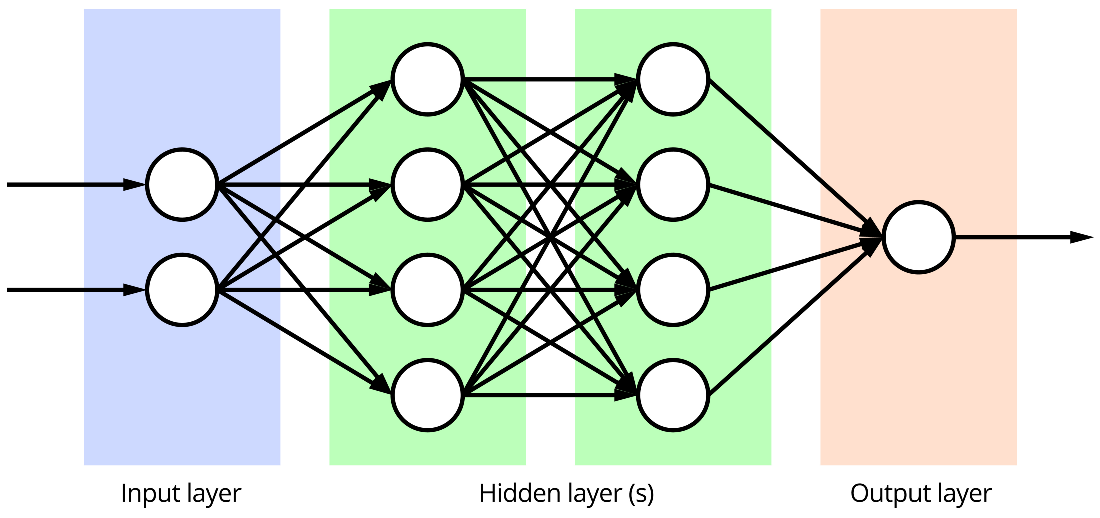

Figure 3: Basic Architecture of a Neural Network

Nodes or neurons in a neural network play a central role in processing information. They receive input, apply activation functions, and transmit the processed data to the next layer, allowing the network to learn complex patterns and relationships.

The network can process various types of data, depending on the problem at hand. The input layer typically receives the raw input features, while the output layer generates the network's prediction. The hidden layers, through their interconnected nodes, process the information by performing complex transformations and feature extractions.

3.4.2 Backpropagation and Gradient Descent

Backpropagation is a crucial technique used to compute gradients of the loss function concerning the network's weights. It's vital for the optimization process in training a neural network.

The weights and biases in a neural network represent the strength of connections between neurons and the neuron's sensitivity to specific input, respectively. Activation numbers depict the output generated by each neuron after processing input data.

While delving into the mathematical details of these concepts is beyond the scope of this section, it's important to understand the high-level principles of how neural networks operate. [3]

3.5 Convolutional Neural Networks (CNNs)

CNNs are a class of deep neural networks, most commonly applied to analyzing visual imagery. They are particularly well-suited for image recognition and classification tasks due to their ability to capture spatial hierarchies in the data.[4]

3.5.1 CNN Architecture

A typical CNN architecture consists of convolutional layers, pooling layers, and fully connected layers. The convolutional layers apply a series of learnable filters to the input, capturing spatial features.

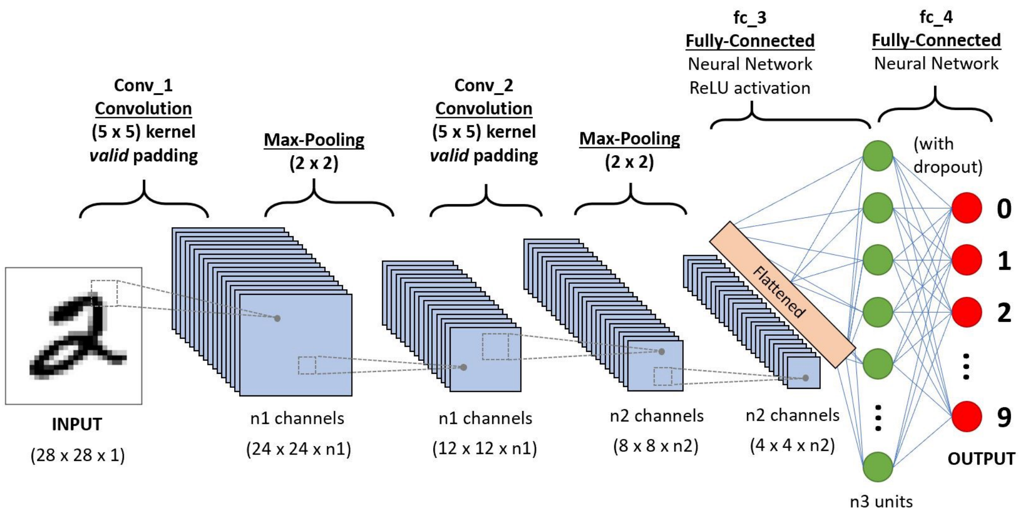

Figure 4: Typical Architecture of a Convolutional Neural Network
Fuente: Di Guan. (2020). Classical Architectures in CNN. Editor 1. `https://guandi1995.github.io/Classical-CNN-architecture/`

3.5.2 Feature Learning in CNNs

CNNs automatically and adaptively learn spatial hierarchies of features from input images. This feature learning aspect makes CNNs particularly effective for image processing tasks.

3.6 Image Processing in Machine Learning

Processing images to be fed into machine learning models involves several steps like resizing, converting to grayscale, binarization, and transforming them into a matrix or vector format.[5]

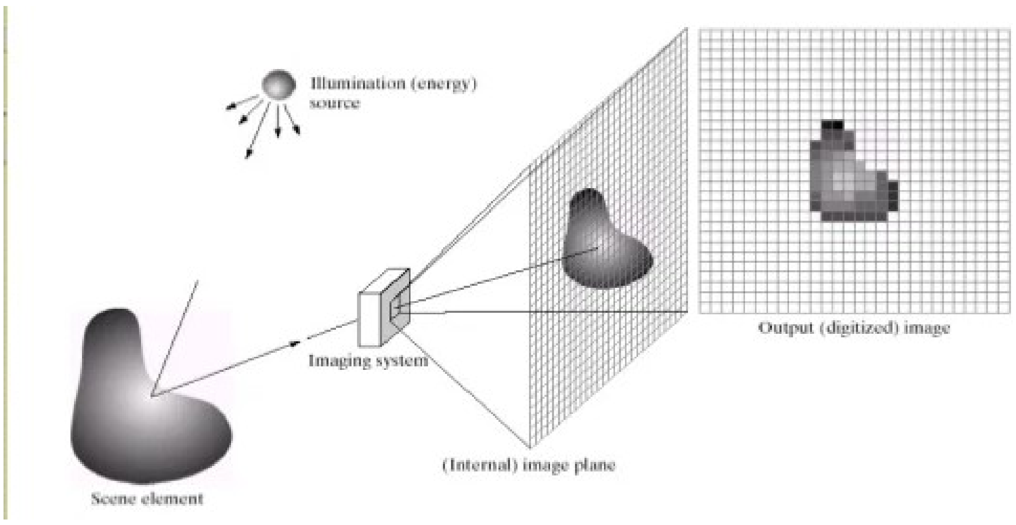

Figure 5: Example of Image Processing Steps
Fuente: Silicon ,M. (2016). Computer Science Thesis in Digital Image Processing. Editor 1. `https://es.slideshare.net/SiliconMentor/computer-science-thesis-in-digital-image-processing#5`

4 Methodology

This section outlines the systematic approach adopted in this project to achieve the objectives delineated earlier, focusing on the development and integration of Multinomial Logistic Regression and Convolutional Neural Networks (CNNs).

4.1 Data Preprocessing

Data preprocessing is a critical step in ensuring that the machine learning models receive high-quality, relevant data for training and prediction.

- **Image Conversion to Grayscale and Binarization:** Images are first converted to grayscale and then binarized to reduce complexity and focus on the essential features.
- **Image Transformation:** The preprocessed images are transformed into a matrix or vector format, making them suitable for feeding into the machine learning models.

4.2 Development of Multinomial Logistic Regression Model

The Multinomial Logistic Regression model is developed to handle the multi-class classification task of digit recognition.

- **Model Training:** The model is trained using a labeled dataset of digit images, where it learns to associate specific features of the images with their corresponding digit labels.
- **Parameter Optimization:** Various parameters, such as the regularization strength and the solver used, are fine-tuned to optimize the model's performance.

4.3 Development of Neural Network

A Neural Network, particularly a CNN, is developed for more advanced digit recognition tasks.

4.3.1 CNN Architecture Design

A CNN is designed with convolutional layers, pooling layers, and fully connected layers tailored to identify complex patterns in the digit images.

4.3.2 Training and Tuning

The CNN is trained on a substantial dataset, with parameters like the learning rate, number of filters in convolutional layers, and number of neurons in fully connected layers being meticulously tuned.

4.4 Integration with Flask

Flask, a Python web framework, is used to create a user-friendly web interface for digit recognition.

4.4.1 Routing and Request Handling

Routing functions are established to handle various web requests, and mechanisms are implemented to process uploaded images or manually entered digits, passing them to the models for prediction.

4.5 Model Evaluation and Validation

Comprehensive evaluation of both models is conducted using metrics such as accuracy, precision, recall, and the F1 Score.

- **Performance Comparison:** The performance of the Multinomial Logistic Regression model and the CNN is compared to determine their effectiveness in digit recognition.
- **Validation Techniques:** Techniques like cross-validation and confusion matrices are employed to validate the models and ensure their robustness and reliability.

5 Resultados

6 Discusiones

References

[1] Hastie, T.; Tibshirani, R.; Friedman, J. (2009). *The Elements of Statistical Learning.* Springer. ISBN 978-0-387-84857-0.

[2] Agresti, A. (2002). *Categorical Data Analysis.* Wiley-Interscience. ISBN 0-471-36093-7.

[3] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.

[4] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

[5] Gonzalez, R.C., Woods, R.E. (2008). *Digital Image Processing* (3rd ed.). Prentice Hall. ISBN 978-0-13-168728-8.

7 Appendix

7.1 Multinomial Logistic Regression

Multinomial logistic regression extends binary logistic regression to handle multiple classes in classification problems.

7.1.1 General Formula

The formula calculates the probability of an instance belonging to class i:

$$p_i = \frac{e^{z_i}}{\sum_{i=1}^{C} e^{z_i}} \tag{1}$$

This probability distribution is essential for determining the likelihood of the instance belonging to each class.

7.1.2 Cost Function

The cost function, cross-entropy, quantifies the disparity between predicted and actual values:

$$H(\mathbf{y}, \mathbf{p}) = -\sum_{i=1}^{n} \sum_{k=1}^{K} y_{i,k} \log(p_{i,k}) \tag{2}$$

It plays a vital role in assessing the performance of the model by evaluating the difference between the predicted and true class labels.

7.1.3 One-Hot Encoding

To numerically represent classes, the One-Hot encoding method is used. Each class is encoded into a binary vector with a single active bit corresponding to the class.

7.2 Neural Networks

Neural networks, inspired by brain neurons, are capable of learning intricate patterns in data.

7.2.1 Variable Relationship

The formula computes the total input z_j^l for a neuron in layer l:

$$z_j^l = w_j^l a^{l-1} + b^l \tag{3}$$

This calculation determines the weighted sum of the inputs from the previous layer along with the bias for the neuron.

7.2.2 Activation

The activation a_j^l of the neuron is obtained by applying an activation function σ to z_j^l:

$$a_j^l = \sigma(z_j^l) \tag{4}$$

The activation function introduces non-linearity to the network's output, allowing it to learn complex relationships in data.

7.2.3 Cost Function

The cost function C_0 measures the discrepancy between activations and actual values:

$$C_0 = \sum_{j=0}^{n_c} (a_j^l - y_j)^2 \tag{5}$$

This function quantifies the error between the network's output and the expected values, aiding in understanding the network's accuracy.

7.2.4 Gradient Descent Algorithm

Gradient Descent is a fundamental optimization algorithm used in training neural networks. It aims to minimize the cost function by iteratively adjusting the weights and biases of the network. The algorithm's core concept is based on calculating the partial derivatives of the cost function with respect to the weights and biases. These derivatives indicate the direction and magnitude of change required to reach the optimal solution.

The gradient descent algorithm involves updating the weights and biases in the opposite direction of the gradient (the vector of partial derivatives). This process iterates until convergence, effectively reducing the cost and improving the model's accuracy.

The partial derivatives are calculated as follows:

$$\frac{\partial C_0}{\partial w_j^k} = \frac{\partial z_j^l}{\partial w_j^k} \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial C_0}{\partial a_j^l} \tag{6}$$

$$\frac{\partial C_0}{\partial b_j} = \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial C_0}{\partial a_j^l} \tag{7}$$

$$\frac{\partial C_0}{\partial a_k^{l-1}} = \sum_{i=0}^{n_l-1} \frac{\partial z_j^l}{\partial a_k^{l-1}} \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial C_0}{\partial a_j^l} \tag{8}$$