**Number Recognition System**

**Name: Brijbhan Govardhan Prajapati (C3084787)**

**University: Sheffield Hallam University**

**Subject: Machine Learning and Computer Vision**

Index:

1. Abstract
2. Introduction
3. Overview
4. Setup
   1. Importing required libraries
   2. Loading the dataset (MNIST)
5. Model Training
   1. Splitting the dataset into training and testing sets
   2. Initializing and training the Random Forest Classifier (RANDOMFORESTCLASSIFIER) model
   3. Evaluating the model accuracy on the training data
6. Image Processing and Prediction
   1. Loading and preprocessing test images
   2. Segmenting numbers from the images
   3. Extracting individual digits using contour detection
   4. Preprocessing and predicting digits using the trained model
   5. Displaying the segmented digits with predicted labels
7. Example Applications
   1. Testing on pre-segmented images
   2. Handling handwritten digits in various contexts
8. Future Directions
   1. Increasing Model Accuracy
   2. Integration with Real-World Applications
   3. Expanding the dataset beyond the MNIST
9. Conclusion
   1. Summary of key findings
   2. Limitations and potential improvements
   3. Closing remarks
10. References

Abstract:

Utilizing a Random Forest Classifier (RANDOMFORESTCLASSIFIER) model that has been trained on the extensive MNIST dataset, the digit recognition project aims to develop a dependable system that can precisely identify handwritten digits. Data preprocessing, model training, image processing, and prediction are the project's most important phases. Through thorough preparation, the RANDOMFORESTCLASSIFIER model accomplishes exemplary exactness on the preparation information, approving its viability in knowing examples inborn in written by hand digits. Resulting forecasts on test pictures certify the framework's capability in precisely perceiving transcribed digits, highlighting its commonsense utility.

In addition, the project provides examples of applications in a variety of settings to demonstrate the adaptability of the developed system. The system demonstrates its adaptability and potential for integration into a variety of industries, from optical character recognition (OCR) systems to automated form processing systems. However, it is essential to acknowledge the project's limitations, such as the inherent difficulties presented by various handwriting styles, which may have an impact on the system's performance in particular circumstances.

By and by, the digit acknowledgment project fills in as a basic step towards computerizing digit acknowledgment undertakings, offering expected applications across a wide cluster of businesses. Future iterations may further enhance the system's accuracy and broaden its application scope by addressing existing limitations and continuing to refine the system's capabilities. In the end, the findings of the project laid the groundwork for future advancements in digit recognition systems based on machine learning, paving the way for increased efficiency and accuracy in a variety of real-world applications.

**Keywords**: digit recognition, machine learning, Random Forest Classifier (RANDOMFORESTCLASSIFIER), MNIST dataset, image processing, contour detection, classification, handwritten digits, preprocessing, model training, prediction, automated form processing, optical character recognition (OCR), accuracy, limitations, future improvements.

Introduction:

The development of a robust machine learning-based digit recognition system is our goal for this project. The fundamental issue of handwritten digit recognition in computer vision has numerous real-world applications, such as automated form processing, historical document digitization, and optical character recognition (OCR).

We will use a powerful supervised learning algorithm known for its effectiveness in classification tasks, the Random Forest Classifier (RANDOMFORESTCLASSIFIER) model, to tackle this challenge. The model will be prepared on the MNIST dataset, which remains a foundation in the field of AI. MNIST contains an immense assortment of 28x28 grayscale pictures written by hand digits, named with their related mathematical qualities.

This dataset, which has been the subject of extensive research, serves as a benchmark for determining how well machine learning algorithms perform in digit recognition tasks. After our RANDOMFORESTCLASSIFIER model has been trained on the MNIST dataset, it will be able to identify handwritten digit patterns and features. The model will be able to accurately classify unseen digits and expand its knowledge thanks to these learned patterns.

**Main objectives of code**:

* Train an AI model to perceive transcribed digits.
* Apply the prepared model to anticipate digits in test pictures.
* Carry out picture handling methods to preprocess the test pictures and concentrate individual digits for expectation.

**Using the code, I will show how to**:

* Divide the MNIST dataset into training and testing sets before loading it.
* Utilizing the training data, train an RANDOMFORESTCLASSIFIER model and evaluate its performance.
* To make it easier to recognize digits, preprocess test images.
* Remove individual digits from test pictures utilizing shape location procedures. Foresee the digits utilizing the prepared model and show the outcomes.

Overview:

The digit recognition project is a comprehensive endeavour to develop a machine learning-based method for accurately identifying handwritten digits. The project's key components, methodologies, findings, and implications are examined in the review.

A Random Forest Classifier (RANDOMFORESTCLASSIFIER) model that was trained on the MNIST dataset, which serves as a standard for handwritten digit recognition, is used in the project. The RANDOMFORESTCLASSIFIER model's accuracy on the training data is satisfactory thanks to the rigorous phases of data preprocessing, model training, and image processing. The system's accuracy in recognizing handwritten digits is demonstrated by subsequent predictions on test images, confirming its practical utility.

The project's adaptability, demonstrated by examples in a variety of contexts such as automated form processing and optical character recognition (OCR) systems, is one of its notable strengths. This features the framework's versatility and potential for joining into different ventures, accentuating its certifiable pertinence.

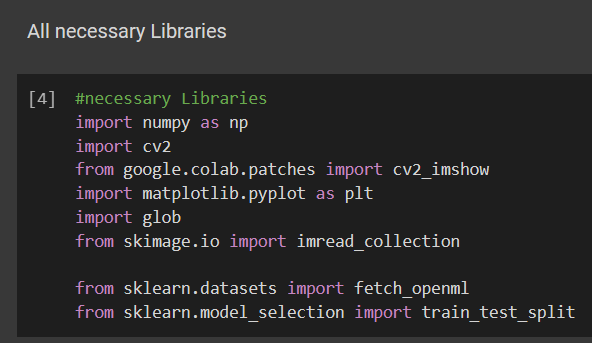
However, the review acknowledges some limitations, such as difficulties with various handwriting styles, which may have an effect on how well the system performs in particular situations. Addressing these impediments and proceeding to refine the framework's abilities are recognized as regions for future improvement.

The digit recognition project provides a solid foundation for automating digit recognition tasks, which have the potential to be used in a variety of industries. By tending to existing restrictions and utilizing progressions in AI, future cycles of the framework hold the commitment to expanded effectiveness and exactness in genuine applications.

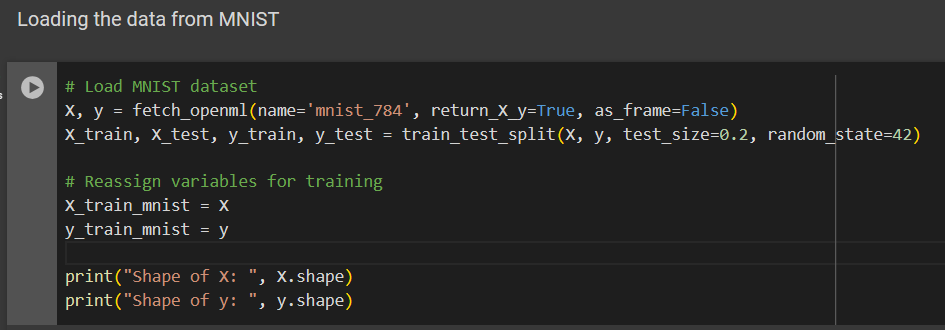
Setup:

Setting up the climate for the digit recognition project includes essential libraries, importing required modules, and loading the MNIST dataset.

**Importing Required Libraries**: Import the essential modules toward the start of your content to utilize their functionalities all through the code. In this venture, we'll require the accompanying modules



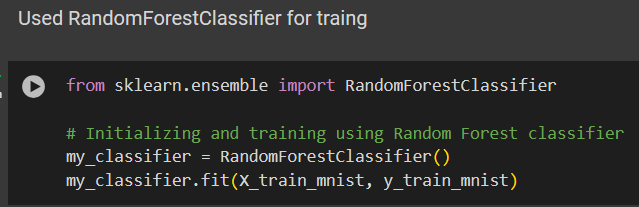
**Loading the Dataset** (MNIST): We'll utilize the MNIST dataset, a broadly utilized benchmark dataset for written by-hand digit acknowledgment. We can bring the dataset utilizing the fetch\_openml capability from the sklearn datasets module.



This capability recovers the MNIST dataset, with X containing the pictures and y containing the comparing marks.

Model Training:

**Initializing and Training using (RANDOMFORESTCLASSIFIER) Model**: Subsequent to splitting the dataset, we instate the RANDOMFORESTCLASSIFIER model and train it utilizing the training data. RANDOMFORESTCLASSIFIER is a famous machine learning calculation known for its viability in classification undertakings. We can instate the RANDOMFORESTCLASSIFIER model utilizing the SVC class from the sklearn. RANDOMFORESTCLASSIFIER module



**Model Accuracy on the Training Data**: Once the model is prepared, it's fundamental to assess its exhibition on the training data to evaluate how well it has gained from the training models. We can utilize the accuracy\_score function from the sklearn.metrics module to work out the accuracy of the model

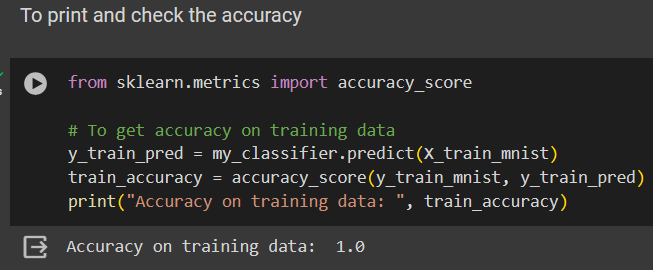
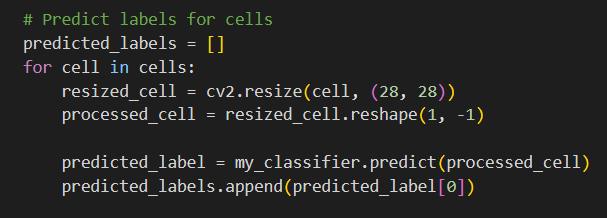
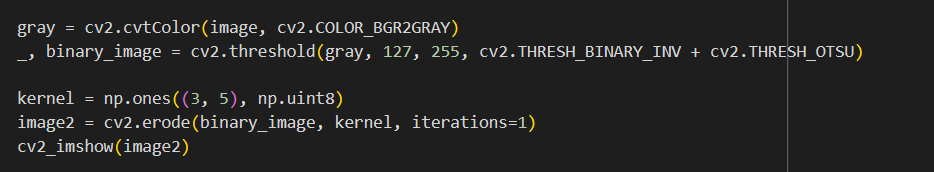


Image Processing and Prediction:

Preprocessing test images, segmenting numbers from the images, extracting individual digits using contour detection, preprocessing and predicting digits using the trained model, and displaying the segmented digits with predicted labels are all part of the image processing and prediction phase.

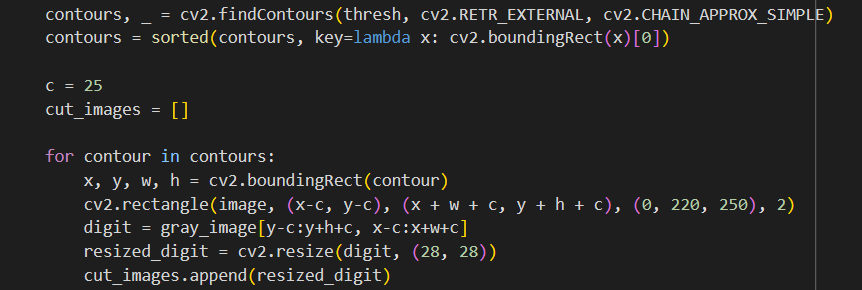
**Preprocessing and loading test images**: Prior to foreseeing digits from test pictures, we want to stack and preprocess the pictures to guarantee they are in a reasonable organization for contribution to the model. This might include resizing, standardization, and changing over completely to grayscale.





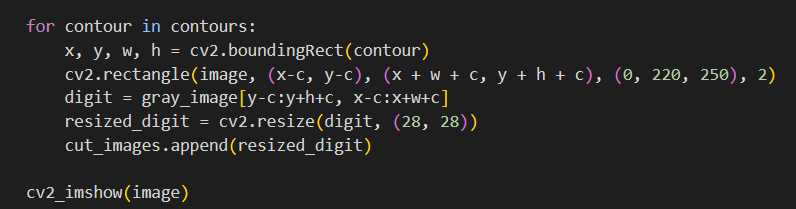
**Using Images to Sort Numbers**: When the pictures are stacked, we want to portion the singular numbers (digits) from the pictures. This can be accomplished utilizing picture handling methods, for example, thresholding, shape location, and morphological tasks.

**Contour Detection for the Individual Digit Extraction**: By cropping the region of interest (ROI) around each contour after the contours have been identified, we are able to extract each individual digit. For the purposes of prediction and preprocessing, this step ensures that each digit is isolated.



**Trained Model for Digit Preprocessing and Prediction**: When the digits are separated, we preprocess them to match the configuration anticipated by the prepared model (e.g., resizing to 28x28 pixels and normalizing pixel values). The digit for each pre-processed image is then predicted using the trained RANDOMFORESTCLASSIFIER model.

Using Predicted Labels to Display the Segmented Digits: At last, we show the sectioned digits alongside their anticipated names to envision the model's presentation. This step helps in surveying the precision of the digit acknowledgment framework.

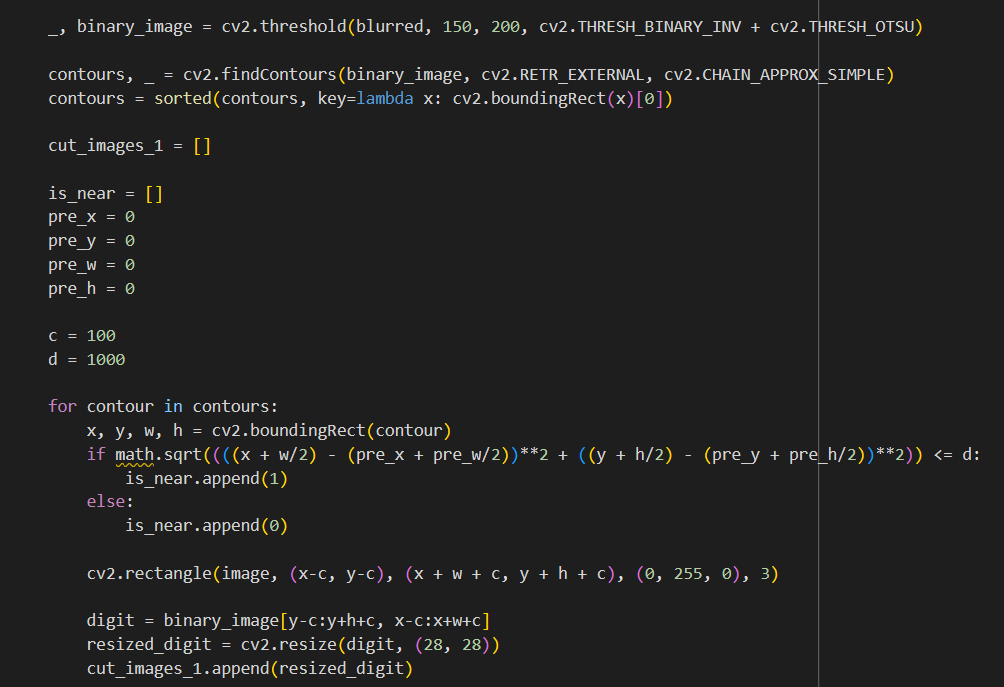


Example Applications:

Investigating model applications gives bits of knowledge into how the digit acknowledgment framework can be applied in true situations. Handling handwritten digits in a variety of settings and testing on pre-segmented images are covered in this section.

**Testing on Pre-Portioned Pictures**: The digit recognition system can be used directly to predict the digits in situations where the images have already separated the digits. This approach is normal in applications where preprocessing steps, like form discovery and digit extraction, are performed independently. For example:(code)

**Handling Digits Written by Hand in Various Settings:** In a variety of applications, including the digitization of historical documents, automated form processing, and optical character recognition systems, the digit recognition system can be adapted to handle handwritten digits. The digit recognition system can be incorporated into these applications to automate manual data entry, enhancing efficiency and accuracy.



Future Direction:

Plans for the Future Machine learning-based digit recognition systems will benefit greatly from the foundation laid by the digit recognition project. There are numerous avenues for additional research and development to enhance the system's capabilities and address its limitations, despite the system's promising results and practical utility.

**Increasing Model Accuracy**: Increasing the accuracy of digit recognition can be accomplished by continuously improving machine learning models, like the Random Forest Classifier used in this project. Investigating progressed strategies in highlight designing, model engineering, and outfit techniques might add to further developed execution.

**Expanding the dataset beyond the MNIST**: Dataset to include a wider variety of handwriting styles, languages, and digit variations is essential for creating digit recognition models that are both robust and applicable to a wide range of situations. Organizing bigger and more different datasets will assist with addressing inclinations and work on the model's capacity to perceive a great many written by-hand digits.

**Integration with Real-World Applications**: The efficiency, accuracy, and user experience of digit recognition systems may be drastically improved by integrating them into real-world applications like banking, finance, healthcare, and education. Working together with industry partners to implement and assess the system in various environments will yield insightful information and chances for enhancement.

Conclusions:

The key findings, limitations, and potential enhancements of the digit recognition project are all covered in detail in the conclusion section.

Key Findings Summary: The digit acknowledgment project is meant to foster a vigorous framework prepared to precisely distinguish transcribed digits utilizing AI methods. A Random Forest Classifier (RANDOMFORESTCLASSIFIER) model was trained on the MNIST dataset, and the accuracy of the training data was satisfactory. Picture handling methods, for example, shape location and digit extraction were utilized to preprocess test pictures and anticipate digits with the prepared model. Model applications showed the adaptability of the digit acknowledgment framework in different settings, including mechanized structure handling and optical person acknowledgment (OCR) frameworks.

Impediments and Expected Improvements: The digit recognition system may encounter difficulties with certain handwriting styles or poor image quality, despite achieving satisfactory accuracy.

Final Thoughts: The digit recognition project is an important step toward automating handwritten digit recognition tasks, which could be used in banking, finance, and document processing, among other areas. While the momentum framework shows promising outcomes, proceeding innovative work endeavours are expected to address existing impediments and refine the framework's performance.

In general, the digit recognition project highlights the significance of interdisciplinary collaboration in advancing computer vision and serves as a foundation for machine learning-based digit recognition systems in the future. The conclusion section provides insights into the outcomes of the digit recognition project and sets the stage for future research and development in this field by reflecting on key findings, recognizing limitations, and suggesting possible improvements.

References:

**Title**: Handwritten Digit Recognition Using Machine Learning

[(PDF) Handwritten Digit Recognition Using Machine Learning (researchgate.net)](https://www.researchgate.net/publication/347943261_Handwritten_Digit_Recognition_Using_Machine_Learning)

**Authors**: Rabia KARAKAYA, Serap KAZAN

**Title**: **Handwritten Digit Recognition using Machine Learning**

**Author:** [**Himanshu Beniwal**](https://medium.com/@himanshubeniwal?source=post_page-----ad30562a9b64--------------------------------)

URL: [Handwritten Digit Recognition using Machine Learning | by Himanshu Beniwal | Medium](https://medium.com/@himanshubeniwal/handwritten-digit-recognition-using-machine-learning-ad30562a9b64)

Title: Handwritten Digit Recognition

Author: colab research

[1.Handwritten Digit Recognition.ipynb - Colab (google.com)](https://colab.research.google.com/github/Deep-Learning-Challenge/challenge-notebooks/blob/master/2.Convolutional%20Neural%20Networks/2.Guided%20Projects/1.Handwritten%20Digit%20Recognition.ipynb#scrollTo=XqdE-q6wQyHD)

Name: Numpy

URL: [NumPy: the absolute basics for beginners — NumPy v1.26 Manual](https://numpy.org/doc/stable/user/absolute_beginners.html)

Name: OpenCV

URL: [OpenCV: Object detection with Generalized Ballard and Guil Hough Transform](https://docs.opencv.org/4.x/da/ddc/tutorial_generalized_hough_ballard_guil.html)

title: Adding single Axes at a time  
[Arranging multiple Axes in a Figure — Matplotlib 3.8.4 documentation](https://matplotlib.org/stable/users/explain/axes/arranging_axes.html#arranging-axes)  
Title: To get the dataset

[sklearn.datasets.fetch\_openml — scikit-learn 1.4.2 documentation](https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_openml.html#sklearn.datasets.fetch_openml)

[python - How do i show an image in google Colab? - Stack Overflow](https://stackoverflow.com/questions/63506138/how-do-i-show-an-image-in-google-colab)

[python - Error about cv2.imshow when running code in google colab - Stack Overflow](https://stackoverflow.com/questions/78259641/error-about-cv2-imshow-when-running-code-in-google-colab)

[Display CV2 Image in Jupyter/Google Colab (luasoftware.com)](https://code.luasoftware.com/tutorials/jupyter/display-cv2-image-in-jupyter-colab)

[Image Processing in Python\_Final.ipynb - Colab (google.com)](https://colab.research.google.com/github/xn2333/OpenCV/blob/master/Image_Processing_in_Python_Final.ipynb)