

A Project Report on  
**Handwritten Digit Recognition**

Submitted by  
**Pratik Kolse (CSE-25)**  
**Shreyash Kawal (CSE-22)**

Under the guidance of  
**Dr. Manju Pawar**

In partial fulfillment of the award of Bachelor of Technology in Computer Science  
and Engineering



**Honor in Data Science under Electronics and Telecommunication Engineering  
Department,  
Maharashtra Institute of Technology,  
Aurangabad (M.S)  
[2022-23]**

## **DECLARATION**

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included; I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

**Place: Aurangabad**

**Date:**

**Signature and Name of students**

**Pratik Kolse**

**Shreyash Kawale**

# **CERTIFICATE**

This is to certify that the minor project report entitled “**Handwritten Digit Recognition**”, submitted by Pratik kolse (CSE 25) & Shreyash Kawale (CSE 22) is the bonafied work completed under my supervision and guidance in partial fulfillment for the award of Bachelor of Technology in Honors & Minors Data Science (ETC Department), Maharashtra Institute of Technology under Dr. Babasaheb Ambedkar Marathwada University, Aurangabad (M.S.).

**Place: Aurangabad**

**Date:**

**Dr. Manju Pawar**  
**Guide**

**Dr. Ajij Sayyad**  
**Head of Department**

**Dr. S. P. Bhosle**  
**Director**  
**Maharashtra Institute of Technology**  
**Aurangabad (M.S.)**

## **APPROVAL CERTIFICATE**

This Minor Project report entitled “**Handwritten Digit Recognition**” by **Pratik Kolse & Shreyash Kawale** is approved for Bachelor of Technology in Honors & Minors Data Science (ETC Department), Maharashtra Institute of Technology under Dr. Babasaheb Ambedkar Marathwada University, Aurangabad (M.S.).

**Place: Aurangabad**

**Date:**

**Examiner:**\_\_\_\_\_

**(Signature)**

\_\_\_\_\_  
**(Name)**

## **ACKNOWLEDGEMENT**

We would like to express special thanks to our beloved Guide **Prof. Dr. Manju Pawar** for his valuable suggestions and rare insights, constant encouragement and inspiration throughout the Project Work.

We express our deep sense of gratitude to our beloved Principal, **Dr. S. P. Bhosle**, for his valuable guidance and for permitting us to carry out this Project.

We would like to take this opportunity to thank **Dr. Ajij Sayyad**, Head of the Department, for providing her great support in successful completion of our Project.

We are grateful to our project coordinator **Prof. Dr. Manju Pawar**, and thanks to all teaching and nonteaching staff members who contributed for the successful completion of our project work.

**Pratik Kolse (CSE-25)**

**Shreyash Kawale (CSE-22)**

## INDEX

<b>1. INTRODUCTION</b>	<b>Page</b>
1.1 Introduction	9
1.2 Necessity	9
1.3 Problem definition	10
1.4 Objectives	10
1.5 Scope	10
1.6 Applications	11
<b>2. TECHNOLOGIES</b>	
2.1 Google Jupyter Notebook	12
2.2 Python	12
2.3 NumPy	14
2.4 Pandas	14
2.5 Matplotlib	15
<b>3. SYSTEM DEVELOPMENT</b>	
3.1 Dataset	16
3.2 Literature Survey	17
3.3 Architecture	18
3.4 Methodology	20
<b>4. CONCLUSION</b>	26
<b>5. REFERENCES</b>	27

## Figures

### List of Figures

Figure	Illustration	Page
3.1	Sample images of MNIST dataset	16
3.2	Architecture	18
3.3	Flow of Training module	20

## Screenshots

<b>Sr. No</b>	<b>Screenshot No</b>	<b>Screenshot Title</b>	<b>Page No</b>
1	Screenshot 1	Library Import	20
2	Screenshot 2	Generating dataset	21
3	Screenshot 3	Data Read	21
4	Screenshot 4	Load the dataset	22
5	Screenshot 5	Preview the Training Output	22
6	Screenshot 6	Training Output	23
7	Screenshot 7	Calculate Accuracy	23
8	Screenshot 8	Prediction of Input Image	24
9	Screenshot 9	Final Output	24
10	Screenshot 10	Output Image	25



# INTRODUCTION

## **Introduction:**

The project is to take a picture of a character and process it up to recognize the image of that character like a human brain recognize the various digits. The project contains the deep idea of the Image Processing techniques and the big research area of machine learning and the building block of the machine learning called Neural Network. There are two different parts of the project

- 1) Training part
- 2) Testing part.

Training part comes with the idea of to train a child by giving various sets of similar characters but not the totally same and to say them the output of this is “this”. Like this idea one has to train the newly built neural network with so many characters. This part contains some new algorithm which is self-created and upgraded as the project need.

The testing part contains the testing of a new dataset. This part always comes after the part of the training. At first one has to teach the child how to recognize the character .Then one has to take the test whether he has given right answer or not. If not, one has to train him harder by giving new dataset and new entries. Just like that one has to test the algorithm also.

## **Necessity:**

The total project lies with a great computation speed and by a online server where run and compilation done quickly. All the packages were imported that were needed for the software online. We need the tools to be imported also.

This project at first is in need of the software of python. The total code is written in python so it needs Python3. Python2 was not chosen because python3 has some additional upgrade over python2. The packages have been imported and the algorithm created which is done by installing the new packages from online in python3.

Apart from that the total project is online compiled or ran and done by the software provided by the Google Colab free version. Apart from choosing Anaconda Navigator, Spyder or Jupyter Notebook, Google Colab or Colabotory have been chosen because this provide more speed and accurate compilation as is known. Creation of the machine learning algorithms deals with data and bigger size programs.

## **Problem Definition:**

The total world is working with the various problems of the machine learning. The goal of the machine learning is to factorize and to manipulate the real-life data and the real life part of the human interaction or complex ideas or the problems in the real life. The most curious of those is Handwritten Character Recognition because it is the building block of the human certified and the classification interaction between other humans.

So, the goal was to create an appropriate algorithm that can give the output of the handwritten digit by taking just a picture of that character. If one asks about Image processing then this problem can't be solved because there can be a lot of noises in that taken image which can't be controlled by human. The main thing is when human write a handwritten character or for our case digit he has no single idea whether he has to draw it in the circulated pixels or just same as a standard image given. A machine can do that but not the human. So by matching only the pixels one can't recognize that.

## **Objectives:**

The primary objective of this project is to develop and implement a sentiment analysis system specifically tailored for analyzing product reviews.

The key objectives include:

Collecting a comprehensive dataset of product reviews from various sources.

Preprocessing and cleaning the dataset to ensure data quality and remove noise.

Training and fine-tuning machine learning or deep learning models for sentiment classification.

Evaluating the performance of the sentiment analysis system using appropriate metrics.

Implementing the sentiment analysis system in a user-friendly interface for practical application.

## **Scope:**

Handwritten Digit Recognition is used in car board system depending on information of the mentality of the driver can be provided to the system to initiate his/her and the customer safety.

☐ Development of a facial emotion recognition system implementing the computer visions and enhancing the advanced feature extraction and classification.

☐ This system can be used in digital in security systems which can identify a person in any form of expression he presents himself.

## **Applications:**

**The applications of sentiment analysis on product reviews are diverse and impactful:**

**Customer feedback analysis:** Businesses can gain valuable insights into customer sentiment, allowing them to address customer concerns, improve products, and enhance customer satisfaction.

**Brand reputation management:** Sentiment analysis can help monitor and manage brand reputation by identifying negative sentiments early on, enabling timely responses and reputation damage control.

**Competitive analysis:** By analyzing product reviews, businesses can compare their products with competitors, identify areas for improvement, and gain a competitive advantage in the market.

**Marketing and advertising:** Sentiment analysis provides valuable input for targeted marketing campaigns, enabling businesses to tailor their messaging to resonate with customer sentiments effectively.

**Product development:** By understanding customer sentiment, businesses can make informed decisions about product enhancements, new features, and future product roadmap.

# TECHNOLOGIES

## Google Jupyter Notebook:

Jupyter Lab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality

By using Google Jupyter Notebook, you can harness the full power of popular Python libraries to analyze and visualize data. The code cell below uses numpy to generate some random data, and uses matplotlib to visualize it. To edit the code, just click the cell and start editing.

## Python:

Python is a programming language widely used by Data Scientists. Python has in-built mathematical libraries and functions, making it easier to calculate mathematical problems and to perform data analysis. Python is in demand for the past few years and the recent survey also suggested the same, Python leads the chart among the top programming languages in both the TIOBE index & PYPL Index. Python is an extremely popular programming language in the field of data science, and it offers several benefits for data scientists. Here are some key advantages of using Python in data science:

**Ease of Use:** Python has a simple and readable syntax, making it easy for beginners to learn and use. Its simplicity allows data scientists to quickly prototype, experiment, and iterate on their ideas.

**Vast Ecosystem of Libraries:** Python has a rich ecosystem of libraries specifically built for data science and machine learning, such as NumPy, Pandas, Matplotlib, and scikit-learn. These libraries provide ready-to-use functions and tools for tasks like data manipulation, analysis, visualization, and model development

**Flexibility and Versatility:** Python is a versatile language that can be used for a wide range of data science tasks. It supports various paradigms, including procedural, object-oriented, and

functional programming. This flexibility allows data scientists to implement complex algorithms and models with ease

**Strong Community Support:** Python has a large and active community of data scientists, researchers, and developers. This means that you can easily find help, resources, and libraries for almost any data science task.

**Integration Capabilities:** Python can seamlessly integrate with other languages and tools, making it easy to leverage existing code and infrastructure.

**Data Visualization:** Python provides powerful libraries like Matplotlib, Seaborn, and Plotly, which enable data scientists to create visually appealing and informative plots, charts, and graphs. These libraries facilitate the exploration and communication of data insights effectively

**Scalability:** While Python itself is an interpreted language, there are frameworks like PyTorch and TensorFlow that allow for efficient execution on GPUs and distributed systems.

**Job Opportunities:** Python's popularity in data science has led to a significant demand for Python-skilled data scientists.

Overall, Python's simplicity, extensive libraries, and strong community support make it an excellent choice for data scientists, allowing them to efficiently tackle various data-related challenges and deliver impactful insights and models

Python has libraries with large collections of mathematical functions and analytical tools.

In this course, we will use the following libraries:

- **Pandas** - This library is used for structured data operations, like import CSV files, create data frames, and data preparation
- **Numpy** - This is a mathematical library. Has a powerful N-dimensional array object, linear algebra, Fourier transform, etc.
- **Matplotlib** - This library is used for visualization of data.

## **NumPy:**

NumPy (Numerical Python) is a powerful library in Python that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. While NumPy is not directly used for sentiment analysis, it can be a valuable tool in the data preprocessing stage of sentiment analysis for product reviews. Here's how NumPy can be useful

**Efficient Data Storage:** NumPy arrays offer efficient storage and manipulation of numerical data. When dealing with large amounts of text data in sentiment analysis, you can represent the textual reviews as numerical vectors. By converting the text into numerical representations, such as Bag-of-Words or TF-IDF, you can store and process the data more efficiently using NumPy arrays

**Vectorization:** Sentiment analysis often involves performing operations on multiple reviews simultaneously. NumPy allows you to apply mathematical operations on arrays in a vectorized manner, which means the operations are executed efficiently in parallel.

## **Pandas:**

Pandas is another essential library in Python that provides high-performance data manipulation and analysis tools. It is particularly useful in sentiment analysis of product reviews due to its ability to handle structured data efficiently. Here's how Pandas can be beneficial

**Data Loading and Preparation:** Pandas provides easy-to-use functions for loading data from various file formats (e.g., CSV, Excel) into a Data Frame, which is a tabular data structure.

**Data Exploration:** Pandas allows you to explore the data and gain insights about the reviews before performing sentiment analysis.

**Text Preprocessing:** Sentiment analysis often requires text preprocessing steps such as tokenization, removing stop words, stemming or lemmatization, and converting text to lowercase.

**Feature Engineering:** In sentiment analysis, it is common to extract additional features from the text that can improve the performance of sentiment classification models.

## Matplotlib:

Matplotlib is a popular data visualization library in Python that provides a wide range of tools for creating static, animated, and interactive visualizations. While Matplotlib is not specifically designed for sentiment analysis, it can be beneficial in the analysis and visualization of sentiment-related data derived from product reviews. Here's how Matplotlib can be used

**Visualizing Sentiment Distribution:** Matplotlib can be used to plot histograms or bar charts to visualize the distribution of sentiment scores or labels in product reviews. This allows you to understand the overall sentiment distribution and identify any biases or patterns

**Time Series Analysis:** If you have sentiment data over time, Matplotlib can be used to create line plots or area plots to analyze how sentiment changes over different time periods. This helps in identifying trends, seasonal patterns, or any significant shifts in sentiment over time

**Comparing Sentiment Across Categories:** Matplotlib can be utilized to create grouped bar charts or box plots to compare the sentiment scores or labels across different categories or product attributes. This allows you to identify variations in sentiment based on different product features, categories, or any other relevant factors

**Word Clouds:** Matplotlib, in conjunction with libraries like WordCloud or nltk, can be used to generate word clouds to visualize the most frequent words or phrases associated with positive or negative sentiment in product reviews.

**Confusion Matrix:** In sentiment analysis, you often evaluate the performance of sentiment classification models. Matplotlib can help visualize the confusion matrix, which shows the actual sentiment labels versus the predicted labels. By plotting the confusion matrix, you can easily identify misclassifications and evaluate the model's performance

**Interactive Visualizations:** Matplotlib can be combined with libraries like Plotly or mpld3 to create interactive visualizations of sentiment-related data. Interactive plots allow for exploration and drill-down into specific data points, providing a more engaging and insightful way to analyze and present sentiment analysis results

# SYSTEM DEVELOPMENT

## 1. DATASET

Modified National Institute of Standards and Technology (MNIST) is a database which is freely available for handwritten digits and is standard for machine learning algorithms. It is similar to TIDigit which is a database of speech created by Texas Instruments, which tasks in speech recognition. For our project, MNIST dataset is used. In this dataset, the images of digits were taken from a variety of scanned documents in which each image is Grey scaled and of 28\*28 pixels. It uses 60,000 images to train the network and 10,000 images to evaluate hoe accurately the network learned to classify the images. Some of the sample images of the MNIST dataset are shown below.

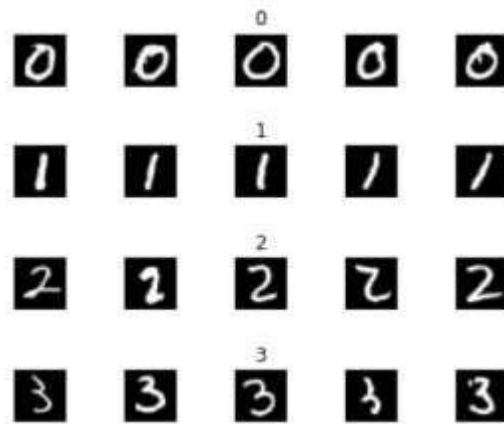


Fig 1: Sample images of MNIST dataset

To use the MNIST dataset in Keras, an API is provided to download and extract images, labels automatically. The task is to classify a given input image of a handwritten digit into one of the 10 classes representing the integer values from 0 to 9 inclusively. The distribution of training data in MNIST is shown below.



## 2. LITERATURE SURVEY

Handwriting recognition of digits has started around the 1980s. The task of handwritten digit recognition, using a classifier, has great significance and it has many applications. Handwritten digits are not similar as they differ in size, thickness, position relative to the margin. These are some of the difficulties we faced while trying to solve this problem. Y. LeCun et al. presented an application of back-propagation networks to handwritten digit recognition

S.no	Title	Author	Algorithm used	Drawbacks	Accuracy(%)
1	The MNIST Database of Handwritten Digits images for Machine Learning Research [4]	Li Deng et al.	Neural network (after distortion)	lack of accuracy due to absence of convolution networks	99.2
2	Deep big simple neural Nets Excel on Handwritten Digit Recognition [11]	Dan Claudiu Ciresan et al.	Simple Neural network and back propagation	Higher processor required, High cost, Time consuming	99.1
3	Digit's recognition using single layer neural Network with principal component analysis [12]	Vineet Singh et al.	PCA Principal component analysis.	Consumes more training time	98.39
4	Handwritten digits recognition using ensemble neural networks and ensemble decision tree [13]	Retno Larasati et al.	Ensemble neural networks that combined with ensemble decision tree	Less accuracy	84
5	Comparison of Classifier methods a case study in handwritten digit recognition [14]	L. Bottou et al.	Baseline Linear Classifier, LeNet 1, Le Net 4, Large fully connected multi network	Much complex networks with high computation time.	1)92.2% 2)98.3% 3)98.9% 4)98.4%

### 3. ARCHITECTURE

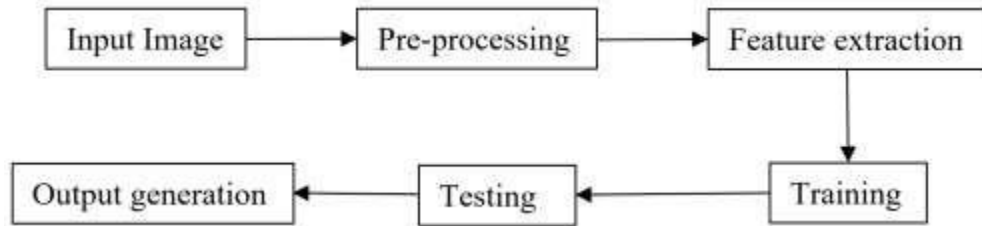


Fig 2: Architecture

The proposed model contains four stages to classify and detect the digits:

#### a) **Pre-Processing:**

Pre-processing is a part of HDR. If there are some rules like a box for each digit then, it will be much easier to detect the boundaries. The fundamental motivation behind pre-processing is to take off noise filtering, smoothing, and standardization. Binarization converts a Greyscaled image into a binary image.

#### b) **Feature Extraction:**

Different type of algorithms used for feature extraction have different types of error rate. The errors made by each separate algorithm does not overlap, so combining all these methods lead to a perfect recognition rate and also helps to reject the ambiguous digits recognition and improve the recognition rate of misclassified digits that can be recognized by humans [10].

#### c) **Classification and Recognition:**

In the classification and recognition step, the extracted feature vectors are given as single input values to each classifier. CNN Convolution layer and the subsampling layer can have various different layers [17]. The down sampling layer is also known as pooling layer [19]. The image is divided into small segments of small areas, and a value is calculated for each area. Then the calculated values are rearranged in sequence to form a new image [7]. This process is similar to fuzzy filter, which can increase the robustness of image feature withdrawal [20]. Extracted features are combined and are defined using the following four

## **Classifiers:**

### **i. K-Nearest Neighbor:**

It is instance-based learning. There are mainly two advantages of using the KNN algorithm, they are, it is robust to noisy training data and it is very efficient if the data is very large in size. In Machine Learning, one of the best ways to learn more about data is by classifying it with what you already know. It is a supervised learning algorithm for multiclass classification.

### **ii. Random Forest Classifier:**

A Random Forest merges a collection of independent decision trees to get a more accurate and stable prediction. It says that there is an immediate connection between the total number of trees and the result it gets. This classifier can be used for regression and also classification. It is a type of ensemble method.

### **iii. Support Vector Machine:**

It finds the hyperplane by performing classifications between the two classes. The most significant advantage of this algorithm is that it provides a regularization parameter which helps in avoiding overfitting problems. It is a classifier that finds an optimal hyperplane that maximizes the margin between two classes.

### **iv. Logistic Regression:**

It is a form of regression where the target variable is binary. What Logistic Regression is great for is an initial benchmark model on a binary classification problem with fairly well-behaved data. It is one of the more transparent algorithms and doesn't work well with really messy data.

### **d) Training and Testing:**

Using the fit () method, a model can be trained. In order to see the skill of the trained model, test data is used as a validation dataset. Finally, to evaluate a model, the test dataset is used. Training is less complex because each module is designed to handle a specific subproblem. It is expected that each module can tackle the specific problem more efficiently and accurately because each module is trained independently which is easy to add and delete modules.

## 4. METHODOLOGY

The goal is to create a model to predict the digit in an image. Steps involved in this project are as follows:

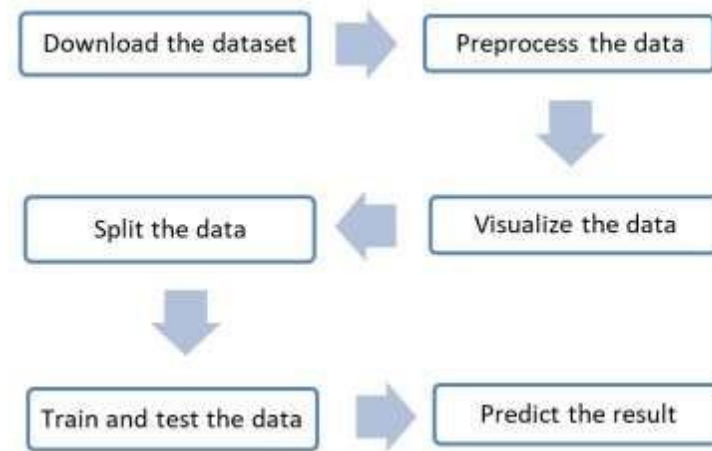


Fig 3: Flow of Training module

The screenshot shows a Jupyter Notebook titled 'Handwritten Character Recognition'. The interface includes a top bar with the Jupyter logo, the title, and a 'Last Checkpoint: 21 hours ago (autosaved)' message. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. To the right of the menu bar are 'Not Trusted' and 'Python 3 (ipykernel)' indicators, along with a 'Logout' button. Below the menu bar is a toolbar with icons for file operations, running, and output. The main area of the notebook contains two input cells. The first cell, labeled 'In [1]:', contains the code `import pyscreenshot as ImageGrab` and `import time`. The second cell, labeled 'In [2]:', contains a loop that iterates from 0 to 99, saving screenshots and clearing the screen. The output of the second cell is visible below the code, showing the execution of the loop for the first 8 iterations (0 to 7).

```
In [1]: import pyscreenshot as ImageGrab
import time

In [2]: images_folder="captured_images/8/"

for i in range(0,100):
    time.sleep(2)
    im=ImageGrab.grab(bbox=(60,170,400,550)) #x1,y1,x2,y2
    print("saved.....",i)
    im.save(images_folder+str(i)+'.png')
    print("clear screen now and redraw now.....")

saved..... 0
clear screen now and redraw now.....
saved..... 1
clear screen now and redraw now.....
saved..... 2
clear screen now and redraw now.....
saved..... 3
clear screen now and redraw now.....
saved..... 4
clear screen now and redraw now.....
saved..... 5
clear screen now and redraw now.....
saved..... 6
clear screen now and redraw now.....
saved..... 7
```

Screenshot 1: Library Import

Jupyter Handwritten Character Recognition Last Checkpoint: 21 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

Generate dataset

```
In [3]: import cv2
import csv
import glob

header = ["label"]
for i in range(0,784):
    header.append("pixel"+str(i))
with open('dataset.csv', 'a') as f:
    writer = csv.writer(f)
    writer.writerow(header)

for label in range(10):
    dirList = glob.glob("captured_images/"+str(label)+"/*.png")

    for img_path in dirList:
        im= cv2.imread(img_path)
        im_gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
        im_gray = cv2.GaussianBlur(im_gray,(15,15), 0)
        roi= cv2.resize(im_gray,(28,28), interpolation=cv2.INTER_AREA)

        data=[]
        data.append(label)
        rows, cols = roi.shape
        ## Add pixel one by one into data array
        for i in range(rows):
            for j in range(cols):
```

Screenshot 2: Generating dataset

Jupyter Handwritten Character Recognition Last Checkpoint: a few seconds ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

```
header.append("pixel"+str(i))
with open('dataset.csv', 'a') as f:
    writer = csv.writer(f)
    writer.writerow(header)

for label in range(10):
    dirList = glob.glob("captured_images/"+str(label)+"/*.png")

    for img_path in dirList:
        im= cv2.imread(img_path)
        im_gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
        im_gray = cv2.GaussianBlur(im_gray,(15,15), 0)
        roi= cv2.resize(im_gray,(28,28), interpolation=cv2.INTER_AREA)

        data=[]
        data.append(label)
        rows, cols = roi.shape
        ## Add pixel one by one into data array
        for i in range(rows):
            for j in range(cols):
                k =roi[i,j]
                if k>100:
                    k=1
                else:
                    k=0
                data.append(k)
            with open('dataset.csv', 'a') as f:
                writer = csv.writer(f)
                writer.writerow(data)
```

Screenshot 3: Data Read

jupyter Handwritten Character Recognition Last Checkpoint: a minute ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

load the dataset

```
In [4]: import pandas as pd
from sklearn.utils import shuffle
data=pd.read_csv('dataset.csv')
data=shuffle(data)
data
```

```
Out[4]:
```

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781
275	8	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
130	1	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
268	8	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
233	8	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
82	0	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
119	0	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
208	8	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
279	8	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1
17	0	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	0	1
98	1	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1

290 rows x 785 columns

Screenshot 4: Load the dataset

jupyter Handwritten Character Recognition Last Checkpoint: an hour ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)


separation of dependent and independent variable

```
In [5]: X = data.drop(["label"],axis=1)
Y= data["label"]
```

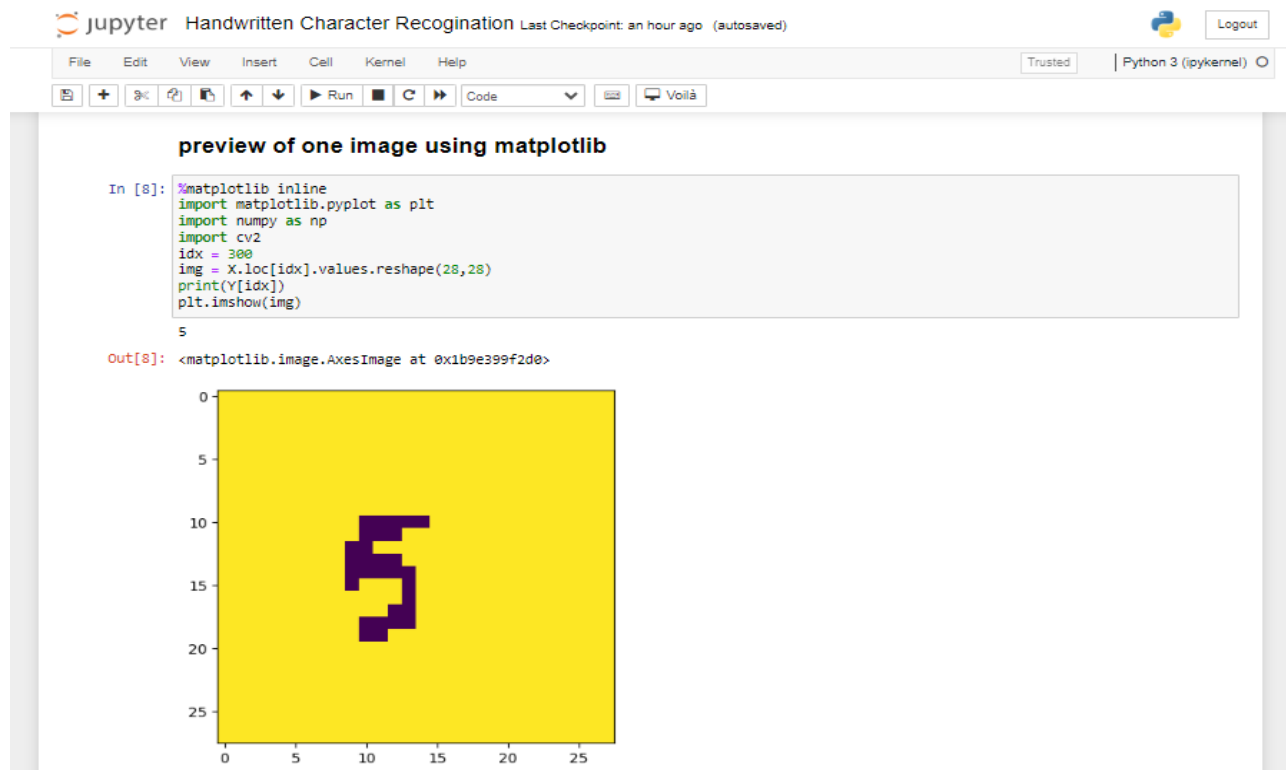
preview of one image using matplotlib

```
In [8]: %matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import cv2
idx = 300
img = X.loc[idx].values.reshape(28,28)
print(Y[idx])
plt.imshow(img)
```

```
Out[8]: <matplotlib.image.AxesImage at 0x1b9e399f2d0>
```



Screenshot 5: Preview the Training Output



Screenshot 6: Training Output



Screenshot 7: Calculate Accuracy

jupyter Handwritten Character Recognition Last Checkpoint: 4 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

prediction of image drawn in paint

```
In [12]: import joblib
import cv2
import numpy as np #pip install numpy
import time
import pyscreenshot as ImageGrab

In [ ]: model=joblib.load("model/digit_recognizer")
images_folder="output/"

while True:
    img=ImageGrab.grab(bbox=(60,170,400,500))

    img.save(images_folder+"img.png")
    im = cv2.imread(images_folder+"img.png")
    im_gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    im_gray =cv2.GaussianBlur(im_gray, (15,15), 0)

    #Threshold the image
    ret, im_th = cv2.threshold(im_gray,100, 255, cv2.THRESH_BINARY)
    roi = cv2.resize(im_th, (28,28), interpolation =cv2.INTER_AREA)

    rows,cols=roi.shape

    X = []

    ## Add pixel one by one into data array
```

Screenshot 8: Prediction of Input Image

jupyter Handwritten Character Recognition Last Checkpoint: 5 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

```
## Add pixel one by one into data array
for i in range(rows):
    for j in range(cols):
        k = roi[i,j]
        if k>100:
            k=1
        else:
            k=0
        X.append(k)

predictions =model.predict([X])
print("Prediction:",predictions[0])
cv2.putText(im, "Prediction is: "+str(predictions[0]), (20,20), 0, 0.8,(0,255,0),2,cv2.LINE_AA)

cv2.startWindowThread()
cv2.namedWindow("Result")
cv2.imshow("Result",im)
cv2.waitKey(10000)
if cv2.waitKey(1)==13: #27 is the ascii value of esc, 13 is the ascii value of enter
    break
cv2.destroyAllWindows()
```

C:\Users\kkite\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but SVC was fitted with feature names  
warnings.warn(

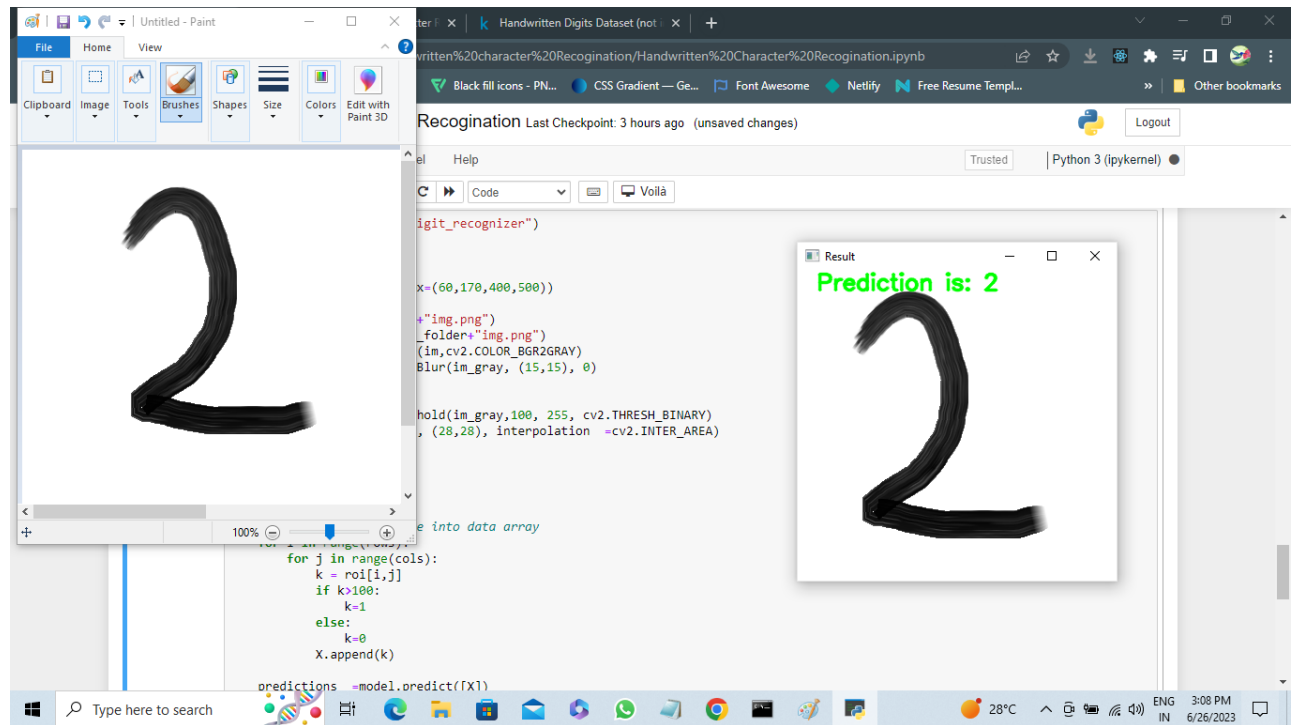
Prediction: 2

C:\Users\kkite\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but SVC was fitted with feature names  
warnings.warn(

Prediction: 2

Screenshot 9: Final Output





**Screenshot 10: Output Image**

## CONCLUSION

Sentiment analysis of product reviews is a valuable project that can provide insights into customer opinions and sentiments towards products or services. By analyzing and classifying the sentiment expressed in reviews, businesses can gain a better understanding of customer satisfaction, identify areas for improvement, and make data-driven decisions to enhance their products or services. The project involves several key steps, including data collection, data preprocessing, sentiment labeling, feature extraction, model development, evaluation, sentiment prediction, post-processing, and continuous monitoring. Throughout these steps, it's important to consider ethical considerations, address potential biases, and ensure the privacy and security of user data. The use of machine learning or deep learning models, along with appropriate preprocessing techniques and feature extraction methods, allows for accurate sentiment classification. Evaluation metrics help assess the performance of the model, while data visualization aids in understanding sentiment patterns and trends.

Implementing sentiment analysis for product reviews provides businesses with actionable insights. They can identify positive and negative aspects of their products, monitor sentiment over time, compare sentiments across different product categories or attributes, and prioritize areas for improvement. This information can inform marketing strategies, customer support, product development, and decision-making processes. It is worth noting that sentiment analysis models are not without limitations. They may struggle with sarcasm, context-dependent sentiment, or nuanced expressions. Therefore, it's important to continuously refine and update the models and consider human validation or other techniques to ensure accuracy and reliability.

Overall, sentiment analysis of product reviews is a powerful tool that empowers businesses to better understand customer sentiments, make data-driven decisions, and improve their products and services based on customer feedback.

## REFERENCE

**Here is a reference list of resources for sentiment analysis of product review projects:**

1. <https://pypi.org/project/opencv-python/>
2. <https://www.tensorflow.org/datasets/catalog/mnist/>
3. <https://stackoverflow.com/questions/21568477/python-opencv-cv2-waitkey-error>
4. <https://www.scribd.com/presentation/306301349/Handwritten-Digit-Regonizer>