```
import cv2
import matplotlib.pyplot as plt
import numpy as np
import operator
import pandas as pd
import random
```

1] Image Processing :

`cv2.imread()` loads an image from the specified file/Path

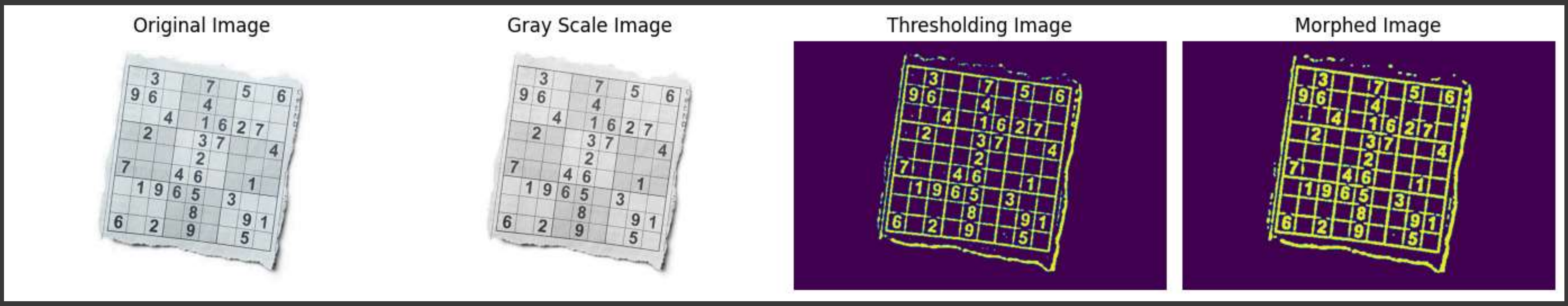
`cv2. cvtColor(,cv2. COLOR_BGR2GRAY)` is used to convert the colour image into Gray Scale

`cv2.adaptiveThreshold()` is used to get values of image array either 0 or 255

`cv2.morphologyEx()` & `cv.dilate()` is used to basically eliminate any unwanted noice present in the image array

Show code

Show code



Original Image

Gray Scale Image

Thresholding Image

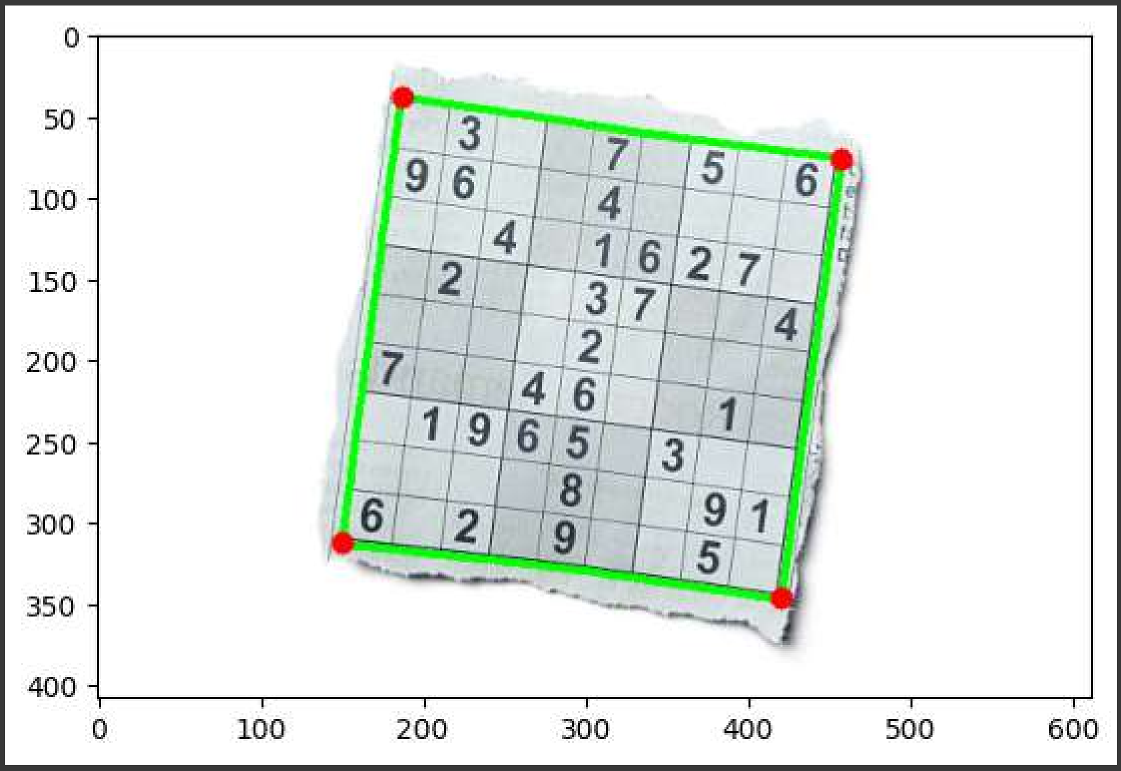
Morphed Image

2] Sudoku Block :

`cv2.findContours()` it detects change in the image color and marks it as contour

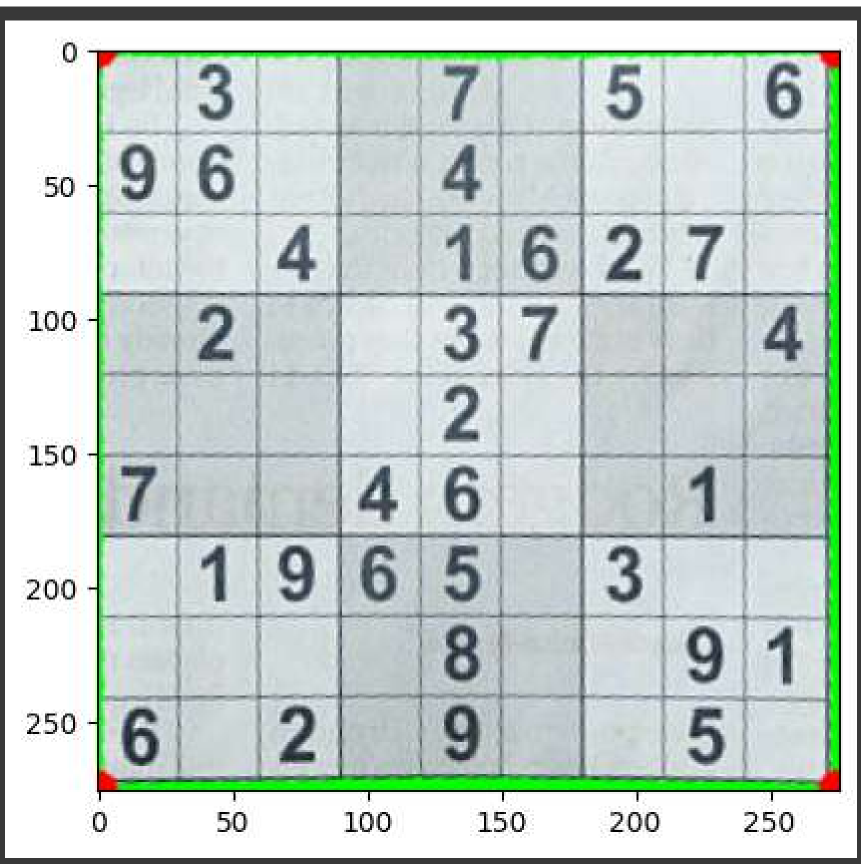
`cv2.arcLength()` & `cv2.approxPolyDP()` to understand Click Here

Show code



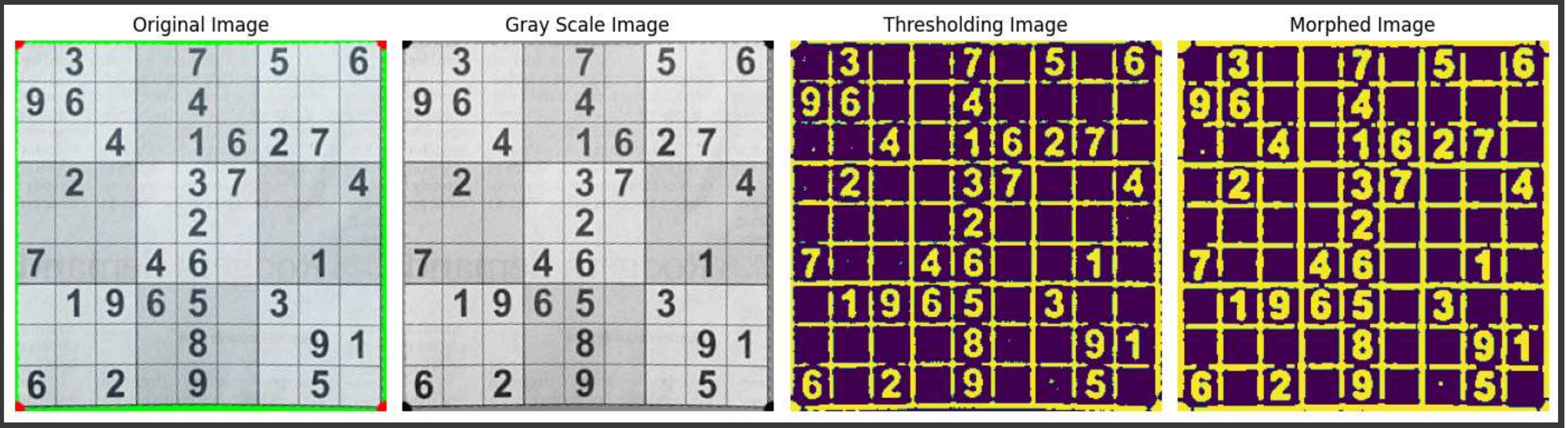
Using `cv2.getPerspectiveTransform()` & `cv2.warpPerspective()` we change the perspective of the image. Through which we end up getting only the sudoku block and eliminating all the other Noice present in the Image.
For better Understanding Click Here

Show code



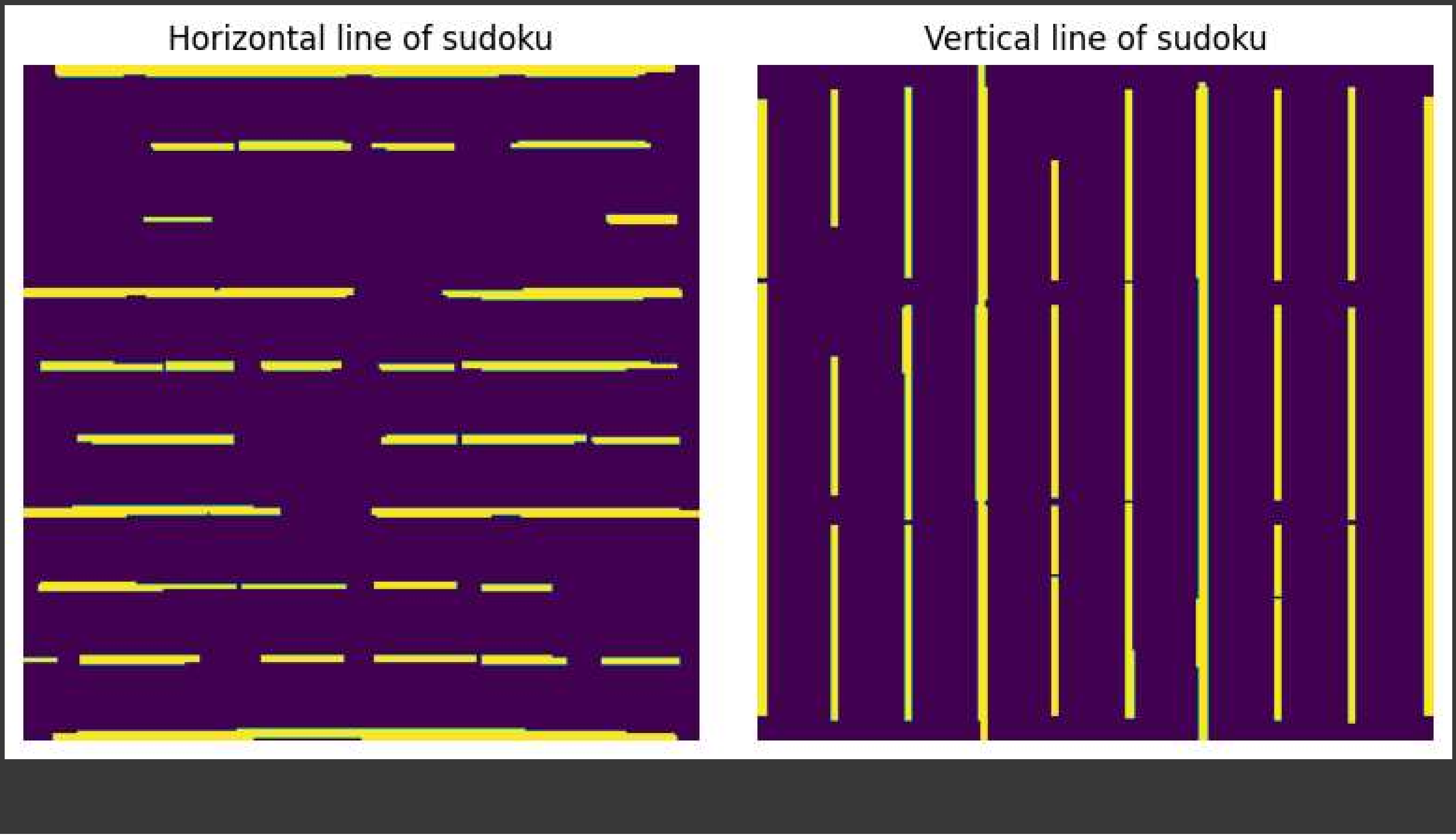
We repeat the pre-processing on this sudoku block

Show code



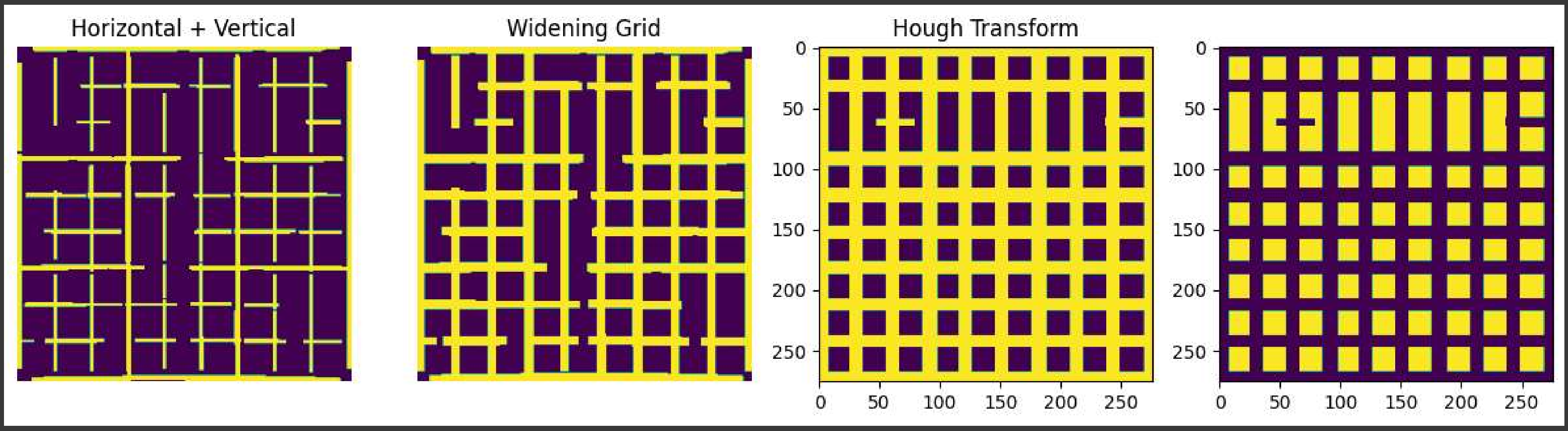
We apply **Morphological Operation** to shapes and structures inside of images.Using it we can increase the size of objects in images as well as decrease them or to close gaps or open them.
`cv2.getStructuringElement()` is a kernel that moves around the image from left to right & top to bottom.
Throught Morphological Operations we extract the vertical and horizontal line running across the Sudoku Block

Show code



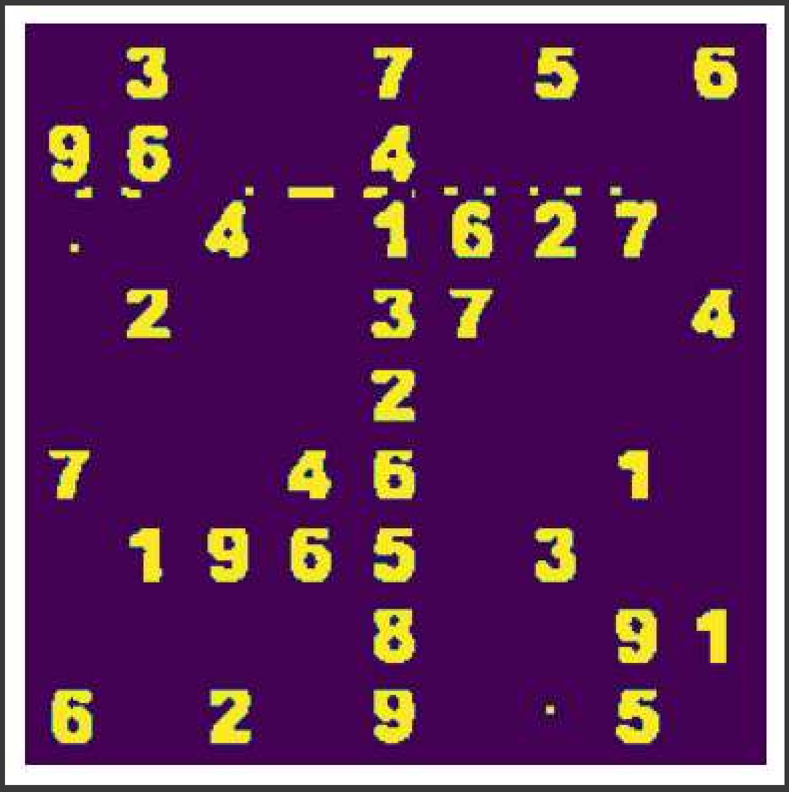
- First this Horizontal and vertical lines are added up to make grid
- Using `cv2.dilate()` we widen the grid
- *Hough Transform** is a popular technique to detect any shape, if it can be represented in mathematical form. It can detect the shape even if it is broken or distorted a little bit. i.e `cv2.HoughLines()`
- Through `cv2.bitwise_not()` we eliminate the grid from the image

Show code



Through `cv2.bitwise_and()` we end up with the image consiting mostly of individual numbers

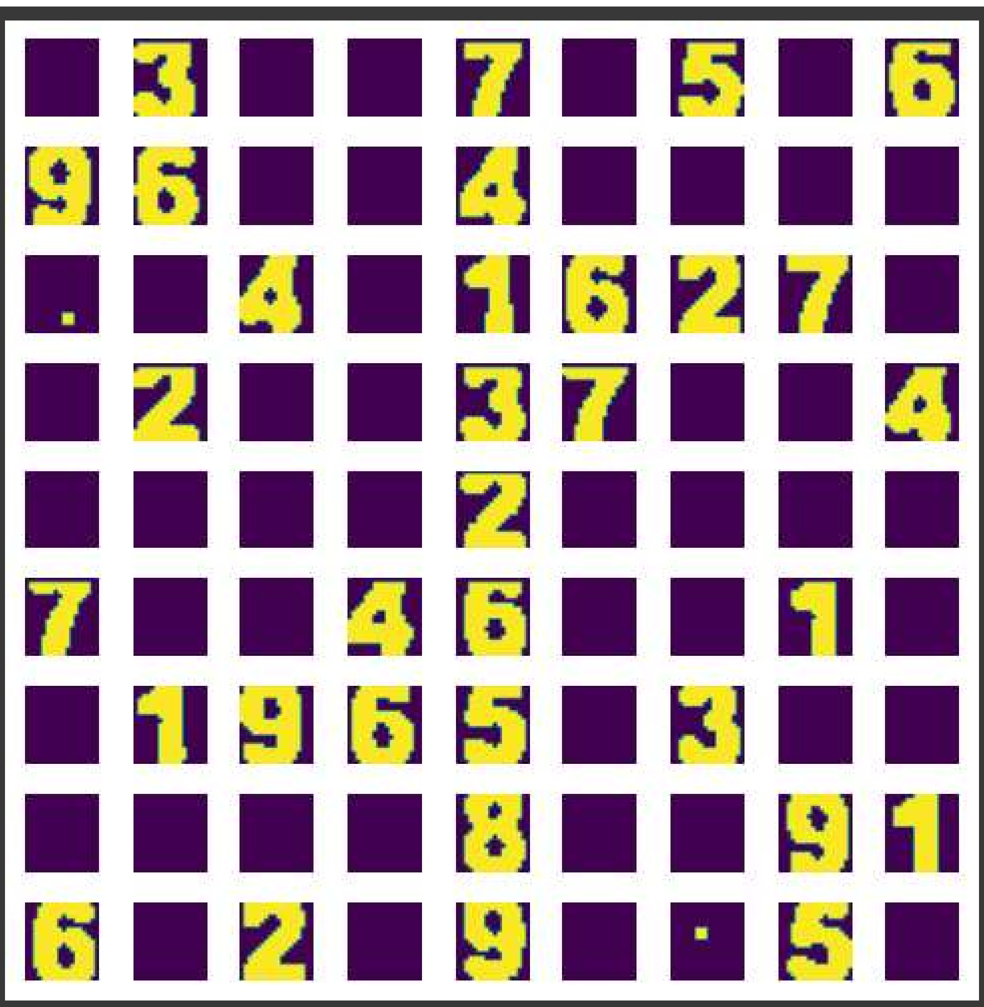
Show code



3] individual Digit :

- The image is sliced into 9 parts vertically and horizontally. i.e 9*9 = 81 images, representing individual 81 blocks present inside the sudoku puzzle
- The individual block from the 81 blocks is given to the model for the prediction and a sudoku puzzle is generated in numpy array format

Show code



	3			7		5		6
9	6			4				
.		4		1	6	2	7	
	2			3	7			4
				2				
7			4	6			1	
	1	9	6	5		3		
				8			9	1
6		2		9		.	5	