SOURCE CODE :

```
import pyscreenshot as ImageGrab
import time
images_folder = "captured_images/0/"
for i in range(0,5):
    time.sleep(8)
    ImageGrab.grab(bbox=(60,170,400,550))
    print("Saved...",i)
    im.save(images_folder+str(i)+'.png')
    print("clear screen and redraw again...")
```

#Generate dataset

```
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):
            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)
```

#load the dataset

```
import pandas as pd
from sklearn.utils import shuffle
data  =pd.read_csv('dataset.csv')
data=shuffle(data)
data
```

#separation of dependent and independent variable

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

#preview of one image using matplotlib

```
%matplotlib inline
```

```
import matplotlib.pyplot as plt
import cv2
idx = 314
img = X.loc[idx].values.reshape(28,28)
print(Y[idx])
plt.imshow(img)
```

#Train-Test split

```
from sklearn.model_selection import train_test_split
train_x,test_x,train_y,test_y = train_test_split(X,Y, test_size = 0.2)
```

#Fit the model using svc and also to save the model using joblib

```
import joblib
from sklearn.svm import SVC
classifier=SVC(kernel="linear", random_state=6)
classifier.fit(train_x,train_y)
joblib.dump(classifier, "model/digit_recognizer")
```

#calculate accuracy

```
from sklearn import metrics
prediction=classifier.predict(test_x)
print("Accuracy= ",metrics.accuracy_score(prediction, test_y))
```

#prediction of image drawn in paint

```
import joblib
import cv2
import numpy as np #pip install numpy
import time
import pyscreenshot as ImageGrab

model=joblib.load("model/digit_recognizer")
```

```
image_folder="img/"

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
   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()

import tkinter as tk
from tkinter import *
from tkinter import messagebox

window=tk.Tk()
window.title("Handwritten digit recognition")

l1=tk.Label(window,text="Digit",font=('Algerian',20))
l1.place(x=5,y=0)

t1=tk.Entry(window,width=20, border=5)
t1.place(x=150, y=0)

def screen_capture():
    import pyscreenshot as ImageGrab
    import time
    import os
    os.startfile("C:/ProgramData/Microsoft/Windows/Start Menu/Programs/Accessories/Paint")
    s1=t1.get()
    os.chdir("E:/DS and ML/Untitled Folder/Untitled Folder/captured_images")
    os.mkdir(s1)
    os.chdir("E:/DS and ML/Untitled Folder/Untitled Folder/")
```

```
    images_folder="captured_images/"+s1+"/"
    time.sleep(15)
    for i in range(0,5):
        time.sleep(8)
        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........")
    messagebox.showinfo("Result","Capturing screen is completed!!")

b1=tk.Button(window,text="1. Open paint and capture the screen",
font=('Algerian',15),bg="orange",fg="black",command=screen_capture)
b1.place(x=5, y=50)

def generate_dataset():
    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):
                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)
    messagebox.showinfo("Result","Generating dataset is completed!!")

b2=tk.Button(window,text="2. Generate dataset",
font=('Algerian',15),bg="pink",fg="blue",command=generate_dataset)
b2.place(x=5, y=100)

def train_save_accuracy():
    import pandas as pd
    from sklearn.utils import shuffle
    data  =pd.read_csv('dataset.csv')
    data=shuffle(data)
    X = data.drop(["label"],axis=1)
```

```
    Y= data["label"]
    from sklearn.model_selection import train_test_split
    train_x,test_x,train_y,test_y = train_test_split(X,Y, test_size = 0.2)
    import joblib
    from sklearn.svm import SVC
    classifier=SVC(kernel="linear", random_state=6)
    classifier.fit(train_x,train_y)
    joblib.dump(classifier, "model/digit_recognizer")
    from sklearn import metrics
    prediction=classifier.predict(test_x)
    acc=metrics.accuracy_score(prediction, test_y)
    messagebox.showinfo("Result",f"Your accuracy is {acc}")

b3=tk.Button(window,text="3. Train the model, save it and calculate accuracy",
font=('Algerian',15),bg="green",fg="white",command=train_save_accuracy)
b3.place(x=5, y=150)

def prediction():
    import joblib
    import cv2
    import numpy as np #pip install numpy
    import time
    import pyscreenshot as ImageGrab
    import os
    os.startfile("C:/ProgramData/Microsoft/Windows/Start Menu/Programs/Accessories/Paint")

    model=joblib.load("model/digit_recognizer")
    images_folder="img/"
    time.sleep(15)
    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
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()


b4=tk.Button(window,text="4. Live prediction",
font=('Algerian',15),bg="white",fg="red",command=prediction)
b4.place(x=5, y=200)


window.geometry("600x300")
window.mainloop()
```