**Comsats University Islamabad, Lahore**

****

|  |  |
| --- | --- |
| **Submitted By** | Sohaib Zahid |
| **Registration No** | FA20-BSE-027 |
| **Group** | B |
| **Session** | 2020-2024 |
| **Submitted To** | Allah Bux Sargano |

***CODE Assignment 3***

***import os***

***import cv2***

***import numpy as np***

***from skimage.feature import hog***

***from sklearn import svm, neighbors, metrics***

***from sklearn.model\_selection import train\_test\_split***

***import tkinter as tk***

***from tkinter import filedialog***

***from PIL import Image, ImageTk # Import PIL for image display***

***# Function to extract HOG features***

***def extract\_hog\_features(image):***

***features, \_ = hog(image, orientations=8, pixels\_per\_cell=(16, 16),***

***cells\_per\_block=(1, 1), visualize=True, channel\_axis=-1)***

***return features***

***# Function to load and preprocess images***

***def load\_images(directory):***

***images = []***

***labels = []***

***label\_mapping = {"500": 0, "1000": 1, "5000": 2}***

***for folder in os.listdir(directory):***

***folder\_path = os.path.join(directory, folder)***

***if not os.path.isdir(folder\_path):***

***continue***

***label = label\_mapping.get(folder)***

***if label is None:***

***continue***

***for file in os.listdir(folder\_path):***

***img\_path = os.path.join(folder\_path, file)***

***if not os.path.isfile(img\_path):***

***continue***

***img = cv2.imread(img\_path)***

***img = cv2.resize(img, (256, 256)) # Increase width and height***

***features = extract\_hog\_features(img)***

***images.append(features)***

***labels.append(label)***

***return np.array(images), np.array(labels)***

***# Load dataset***

***data\_folder = "C:/Users/Sohaib/Desktop/Subjects - Sem 7/CV/Assigmnment 3/augmented-dataset"***

***images, labels = load\_images(data\_folder)***

***# Split dataset***

***train\_imgs, test\_imgs, train\_lbls, test\_lbls = train\_test\_split(images, labels, test\_size=0.2, random\_state=42)***

***# Train SVM***

***svm\_classifier = svm.SVC()***

***svm\_classifier.fit(train\_imgs, train\_lbls)***

***svm\_predictions = svm\_classifier.predict(test\_imgs)***

***svm\_accuracy = metrics.accuracy\_score(test\_lbls, svm\_predictions)***

***# Train KNN***

***knn\_classifier = neighbors.KNeighborsClassifier()***

***knn\_classifier.fit(train\_imgs, train\_lbls)***

***knn\_predictions = knn\_classifier.predict(test\_imgs)***

***knn\_accuracy = metrics.accuracy\_score(test\_lbls, knn\_predictions)***

***# GUI Functions***

***def classify\_image():***

***file\_path = filedialog.askopenfilename()***

***if file\_path:***

***img = cv2.imread(file\_path)***

***img = cv2.resize(img, (256, 256)) # Increase width and height***

***features = extract\_hog\_features(img)***

***svm\_prediction = svm\_classifier.predict([features])[0]***

***knn\_prediction = knn\_classifier.predict([features])[0]***

***currency\_labels = {0: "500 PKR", 1: "1000 PKR", 2: "5000 PKR"}***

***svm\_result = currency\_labels[svm\_prediction]***

***knn\_result = currency\_labels[knn\_prediction]***

***# Display the input image on the GUI***

***img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB) # Convert to RGB format***

***img = Image.fromarray(img)***

***img = ImageTk.PhotoImage(img)***

***img\_label.img = img***

***img\_label.config(image=img)***

***img\_label.image = img # Keep a reference***

***result\_label.config(text=f"SVM: {svm\_result} (Accuracy: {svm\_accuracy\*100:.2f}%), KNN: {knn\_result} (Accuracy: {knn\_accuracy\*100:.2f}%)")***

***result\_label.config(font=("Helvetica", 16), fg="#4CAF50") # Style the result label***

***# Setup GUI***

***root = tk.Tk()***

***root.title("Currency Recognition System")***

***root.geometry("800x600")***

***# Customize the appearance of the GUI***

***root.configure(bg="#333333") # Background color***

***open\_button = tk.Button(root, text="Open Image", command=classify\_image, bg="#4CAF50", fg="white")***

***open\_button.pack(pady=20)***

***open\_button.config(font=("Helvetica", 18)) # Button font and size***

***img\_label = tk.Label(root, bg="#333333")***

***img\_label.pack()***

***result\_label = tk.Label(root, text="", bg="#333333")***

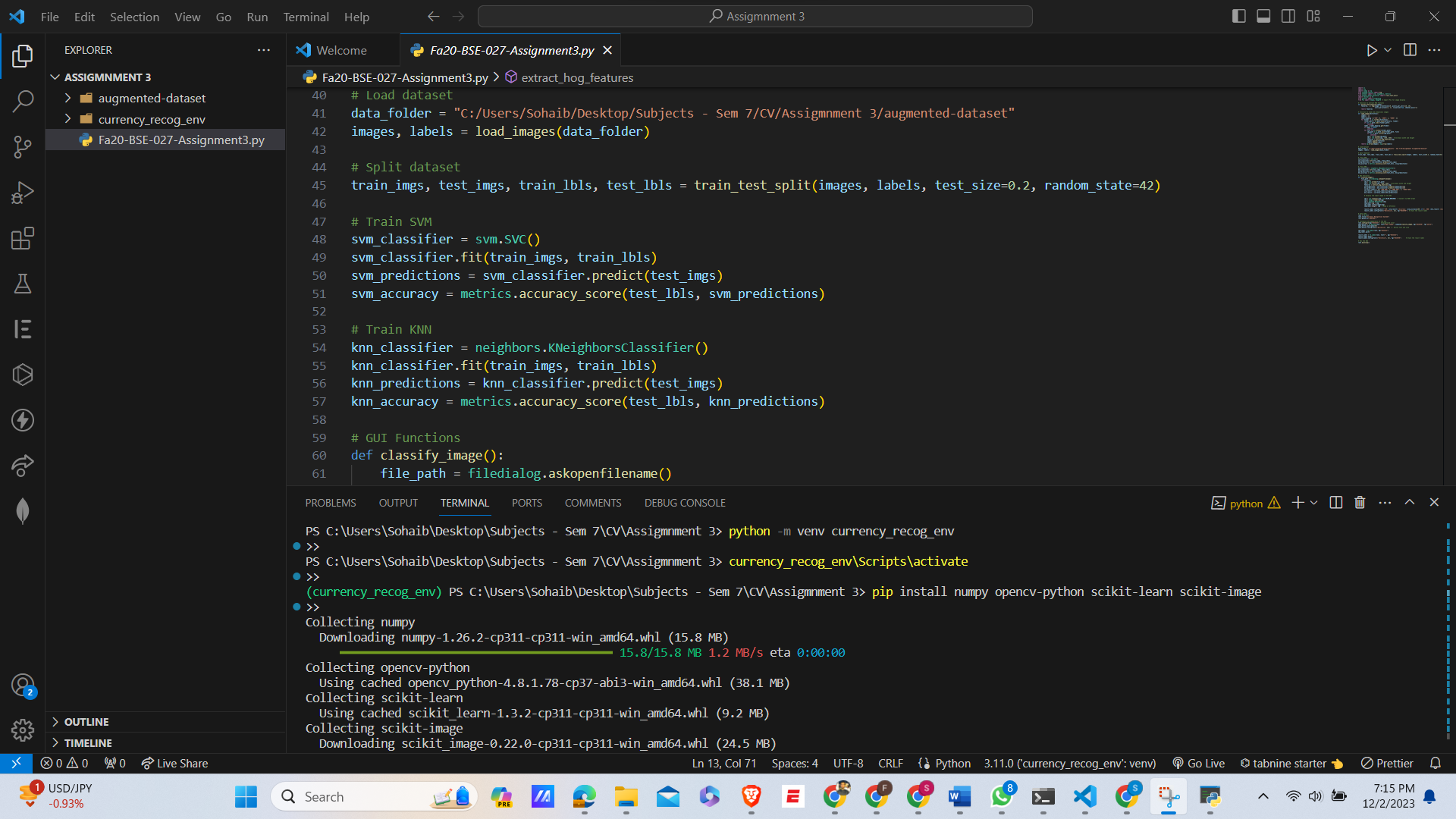
***result\_label.pack()***

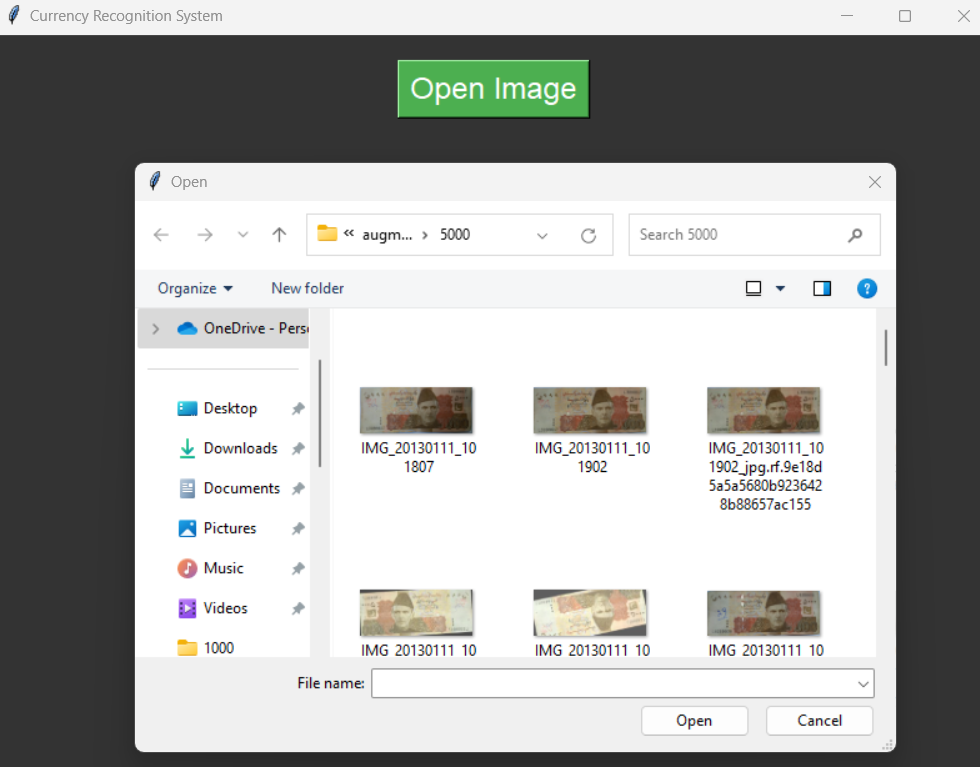
***result\_label.config(font=("Helvetica", 16), fg="#4CAF50") # Style the result label***

***# Run the GUI***

***root.mainloop()***

**Screenshots:**

****



A close up of a currency

Description automatically generated

A screenshot of a computer screen

Description automatically generated