Digital Image Processing Exercise Report

Instructor: MITSUGAMI IKUHISA

Student ID: 81322400098

Student Name: GARCIA AARON CALVIN

Date: 2024 06 25

**Summary/Discussion:**

**Background**

k-Nearest Neighbors is a simple classification algorithm that uses nth-dimensional vectors as inputs, and compares any new examples it receives to the values it has “graphed”, grouped, and classified using training labels.

It operates under a simple decision convention:

* When a scalar distance between two vectors between a testing value and the nearest training example is less than a certain threshold, that testing example is assigned that classification.
* If not “reject” it from that classification... (As it is unrecognizable...)

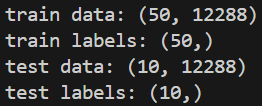
**Preprocessing/Preparation:**  
 I started by reading in the images to the program, and formatting the arrays. I did this by assigning the images and labels from the training and testing datasets, and then I normalized the pixel values of these images by resizing them to 64x64 and flattening them into one-dimensional arrays, each bucket representing a 64th dimensional vector value of each individual image. This normalization allows for the KNN model to intake each image more cleanly. (I also printed out the shapes of the training/test data/labels as to verify that everything was working as intended.)

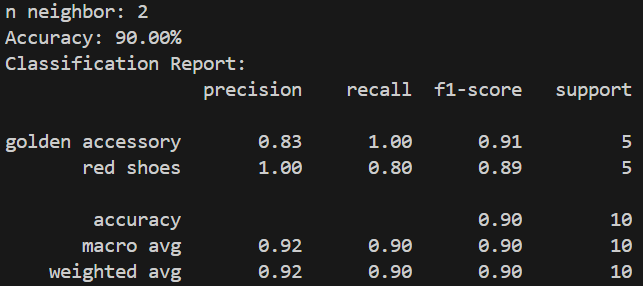
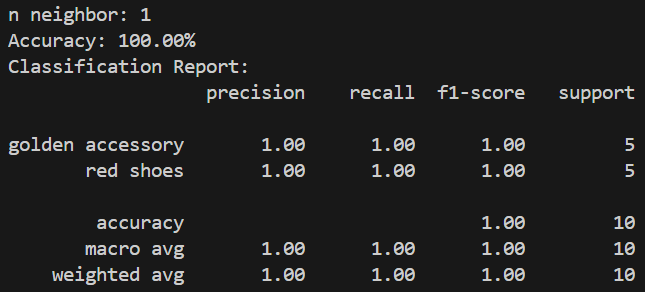
**KNN Testing/Training**  
 My (original) implementation of this classification algorithm is using a for loop that runs the KNN classifier function provided by scikit 5 separate times, each for the currently numbered loop it is on. The “KneighborsClassifier” model uses the index of “i” as each loop’s “k”, and “eucledean distance” as the metric evaluation method. Using the “knn.fit” and “knn.predict” functions, I trained and subsequenctly predicted the training and testing data respectively. After which, I printed the classification report metrics of the model, not before calculating the accuracy with accuracy\_score.

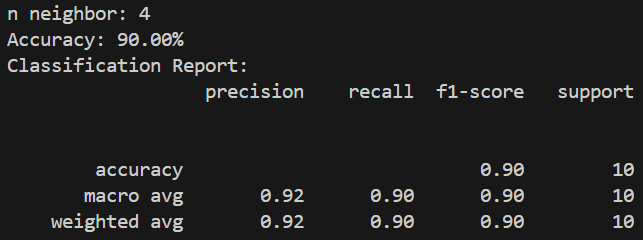
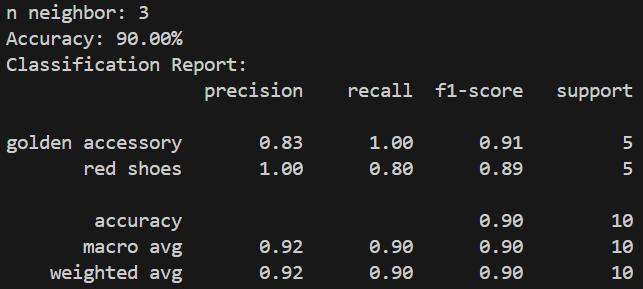
**Results:**

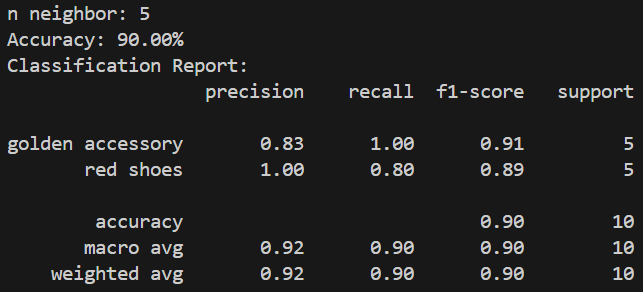
As seen in the provided data section, the entire model had some intriguing results. K=2-5 Practically all had the same values. However, once I turned up k to be a larger number, the accuracy started to slip by a considerable amount.

Provided Data:

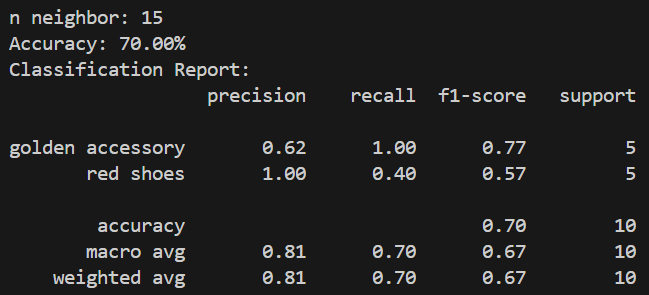
**Classification Reports:**







**Extra Classification:**



**References**

1. Week 10 Slides
2. https://cs231n.github.io/classification/#k---nearest-neighbor-classifier
3. https://docs.opencv.org/3.4/d5/d26/tutorial\_py\_knn\_understanding.html
4. https://docs.opencv.org/3.4/d8/d4b/tutorial\_py\_knn\_opencv.html