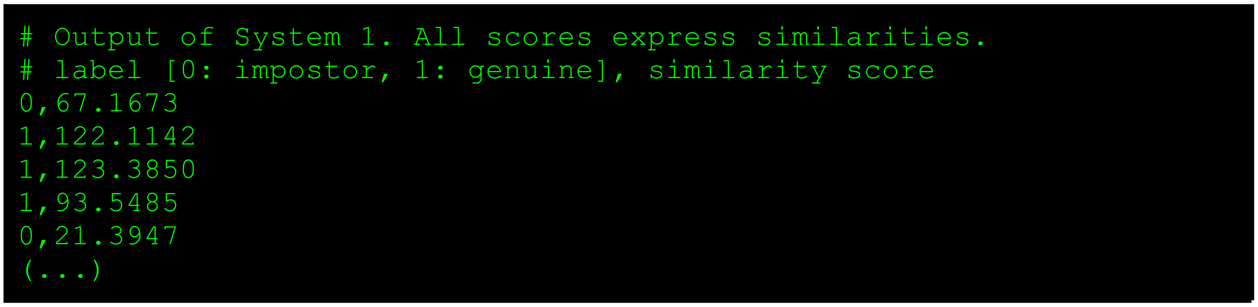
Biometrics (COMP388-002/488-002)
Loyola University Chicago, Fall 2025
Assignment 1: Comparison of Biometric Systems
Due date: Sep 29, 11:59 PM ET
Total: 10 points

1. Introduction

The purpose of this assignment is to train and evaluate the students' abilities to compare the output of different Biometric systems, regardless of the trait they rely on. To do so, three files containing Biometric system outputs are being provided, one for each hypothetical system, namely *s1.csv*, *s2.csv*, and *s3.csv* (https://tinyurl.com/yzewsd94). The *Python* function to load the content of these files into memory was provided in the first coding class (https://tinyurl.com/53knr8y5).

1.1. Input file format

Each input file is a text file containing two header lines (starting with the "#" character, which should be ignored) and 10,000 following lines, one for each response of the respective Biometric system. Lines contain comma-separated values (CSV, hence the ".csv" extension), with one value related to the actual *label* (a.k.a. ground truth) of the output provided by the system, and the other value associated with the *score* computed by the system. We call each of these lines a "*(label, score)*" system *observation*. Figure 1 summarizes the content of *s1.csv*, for illustration.



```
# Output of System 1. All scores express similarities.
# label [0: impostor, 1: genuine], similarity score
0,67.1673
1,122.1142
1,123.3850
1,93.5485
0,21.3947
(...)
```

Figure 1. The first seven lines of file *s1.csv*

Labels are either *0* (for *impostor* trait presentations, such as face presentation followed by a wrong identity claim) or *1* (for *genuine* trait presentations, such as face presentation followed by a correct identity claim). Scores are real numbers comprising either *similarities* or *distances* between the presented trait and the claimed identity template, computed by the respective hypothetical system. Each input file independently defines whether all of its scores are either similarities or distances.

1.2. Assignment directions

To complete this assignment, you will need to access the Google Colab notebook (https://tinyurl.com/53knr8y5) explained in class and make your own copy. After downloading and unzipping the three CSV output files from Sakai to your local computer and uploading them to your Google Colab notebook copy, please follow the instructions and answer the questions presented in Sec. 2.

There is no formal template for providing your answers. You may use the editor you like. The following option should work fine:

- A single PDF file or Word document containing all your answers and generated figures.

Please submit your file through the respective open assignment in Sakai by September 29, 2025, 11:59 PM ET.

2. Questions

Considering the content of the three input files *s1.csv*, *s2.csv*, and *s3.csv*, one for each hypothetical Biometric system, please answer the following questions. You may leverage and adapt the functions and metrics available in the given Google Colab notebook.

2.1. For each one of the three Biometric systems, what score threshold (a.k.a. operating point) should you use? Please explain your answer and describe how you have obtained each one of the respective system thresholds. (1 point)

2.2. For each system, plot and provide a graph with the distributions of its respective scores.
(1 point)

2.3. According to the d' (d-prime) values that one might compute for each system, which of the three should you use if you had to select only one for identification? Please justify your answer. (1 point)

2.4. Plot and provide a single graph with the ROC curves and AUCs of all three systems together. A reference to help: https://tinyurl.com/23fkm3jf. (1 point)

2.5. According to the ROC curves and AUC values, which one of the three systems should you use if you had to select only one for identification? Please justify your answer. (1 point)

2.6. Re-do question 2.4, but this time leveraging the **optimized** *scitkit-learn* (https://en.wikipedia.org/wiki/Scikit-learn) implementation of AUC calculation (see https://tinyurl.com/nsyu2k9b). Which implementation is faster: the naive one presented in class or the one using scikit-learn? Please justify your answer with collected runtimes. (5 points)