2. Questions

Considering the content of both *genuine.txt* (with 577 pairs of fingerprint image file paths) and *impostor.txt* (also with 577 pairs of fingerprints), please answer the following questions.

2.1. For each one of the 1154 available pairs of fingerprint images (577 from *genuine.txt* and 577 from *impostor.txt*), provide the minutiae-based similarity score, as defined in slide 40 of the presentation available at https://bit.ly/3gVJ5N7. To present these scores, generate a single *output.csv* file with 1154 data lines; the first 577 data lines must be respective to the 577 lines of *genuine.txt*, while the following 577 data lines must be respective to *impostor.txt*. Lines with comments must start with "#". The format of this file is explained in Figure 3 through an example, and it follows the same format of the input files used in the first assignment. (4 points)

```
# System output. Line format: label [0: impostor, 1: genuine], score
1,0.6285714285714286
1,0.6046511627906976
(...)
0,0.1326530612244898
0,0.2732919254658385
(...)
```

Figure 3. Expected content for *output.csv*. The scores and number of lines presented here are for mere illustration.

Example file available at https://danielmoreira.github.io/teaching/biometrics-spr22/output.csv.

Code snippets of reading genuine.txt and writing similarity score to file (same was done with impostor.txt, only with 0,...:

```
#generating genuine pairs
with open('fingerprint_data/genuine.txt') as f:
    genuine_array = []
    for line in f:
        pair = line.strip().split(',')
        if len(pair) > 0 and pair[0][0] != '#':
            finger1 = pair[0]
            finger2 = pair[1]

            genuine_array.append((finger1, finger2))
```

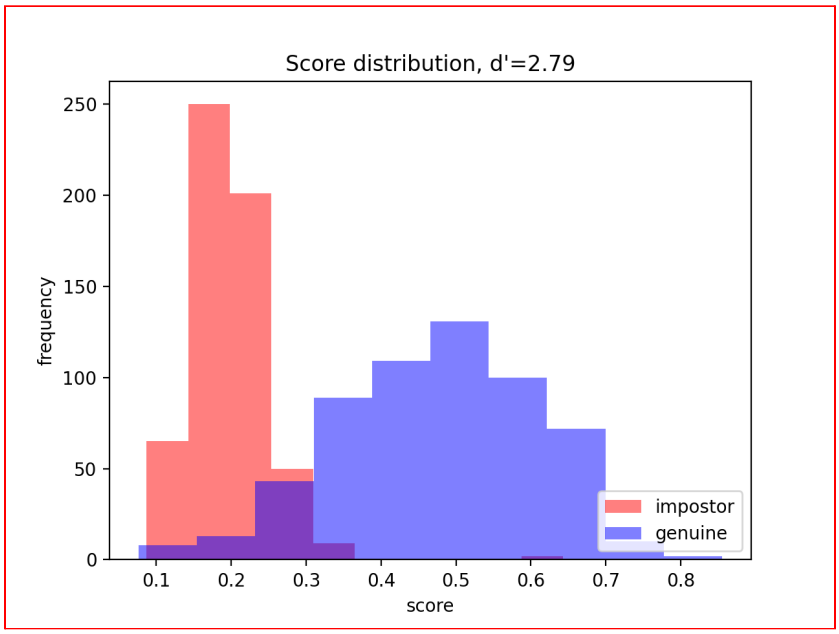
```
f = open("output.csv", "a")
f.write('1,'+ str(len(best_matches)/((len(minutiae_set_1)+len(minutiae_set_2))/2))+'\n')
f.close()
```

2.2. Based on the scores you have obtained, what score threshold (a.k.a. operating point) should you use for this system? Please explain your answer and describe how you have obtained this threshold. (1.5 point)

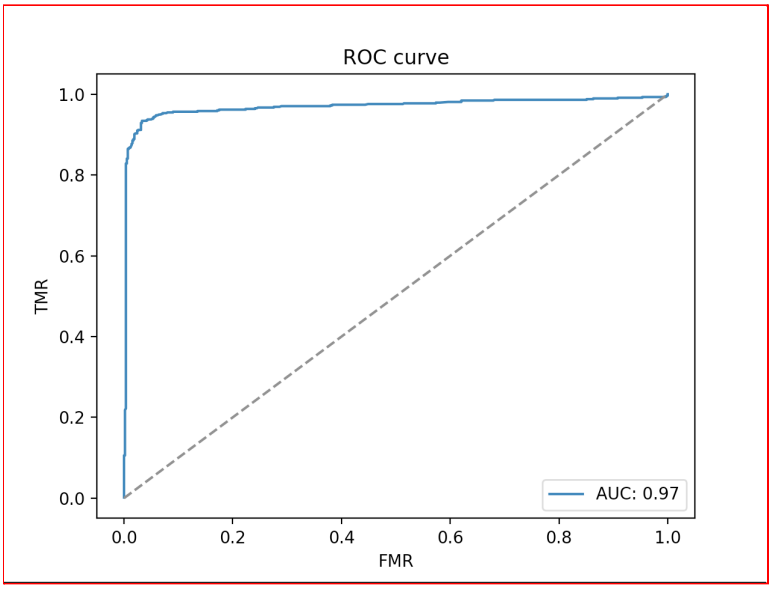
The threshold should be 0.2675. I found this score by using a function (compute_sim_fmr_fnrm_eer) to calculate the equal error rate, which is when the false match rate and false non-match rate are closest to (or ideally, equal to) each other. By using this to calculate the threshold, we avoid showing preference to false matches or false non-matches; rather, we find the threshold where both are low at the same time. When the threshold is 0.2675, the FNRM and FMR are both equal to 0.05546.

2.3. Plot and provide a graph with the distribution of the scores obtained by the system. What is the system's d-prime value? (1.5 point)

The systems D Prime value is 2.7857668953816646.



2.4. Plot and provide a graph with the ROC curve and AUC of the system. Is this system working better than chance? Please explain your answer. (1.5 point)



This system is working significantly better than pure chance. The chance diagonal is the dotted gray line- a curve with less area than the line would be performing worse than chance, where a system has a true 50/50 shot of making an error in either direction.

2.5. In your opinion, would this solution be robust to fake fingerprints such as silicon fingers? Please justify your answer. (1.5 point)

I would say this model is likely not robust enough to detect fake fingerprints such as silicone fingers. The reason is because our system only analyzes the level 2 features of the fingerprint, which are the ridge endings or the ridge bifurcations. Silicone fingers can reasonably replicate these features, but notably would not contain some of the deeper details such as level 3 features. In order to combat against silicone fingers, the model would require a sensor that has a high enough resolution to capture level 3 features, as well as having code that can also identify those features.