COMP30027 Machine Learning Asst1 Report

Liangdongfang Xu 1174154

Task 1. Pop vs. classical music classification

1.
The accuracy is 0.9767441860465116
The precision is 0.9523809523809523
The recall is 1.0

2.
The graphs on the right side (Figure 1) are three density plots of spectral centroid mean, harmony mean and tempo. Each plot consists a blue curve and a red curve, which corresponds to pop and classical respectively.

I will choose spectral centroid mean to be the attribute X. The reason to choose one of the three attributes is that we want one attribute to best distinguish two labels (pop and classical). We don't want to face the situation that some test values of the attribute would result into a dilemma for separating two labels, that is to find the graph which has the minimum area of overlaps between the blue and red curves. By using visual approximation, we will certainly exclude the harmony mean and temp as they apparently have big overlapping.

To roughly calculate the area of overlapping, spectral centroid mean approximately has a triangle (1200*0.0002/2=0.12), harmony mean has a triangle (0.0025*600/2=0.75) and tempo has a triangle (80*0.012/2=0.48). We would like to choose the one with the minimum area, that is spectral centroid mean.

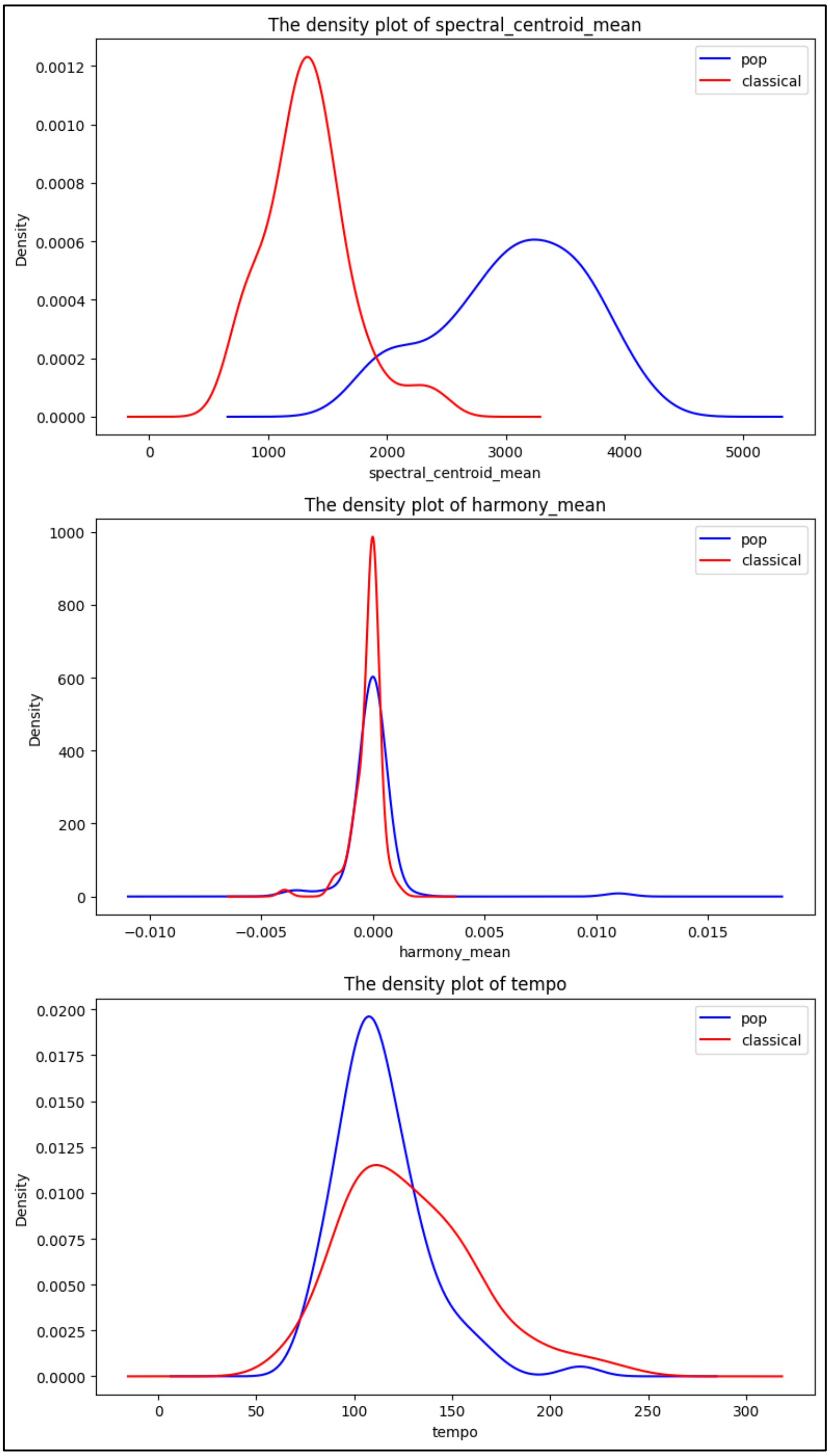

Figure 1

Task 2. 10-way music genre classification

Q6 Dealing with missing values.

My function will randomly generate 20 proportional values from 0 to 1. A function called delete_value_by_proportion will be called to delete some values of attributes of the training data frame based on the proportion given. Then this modified data frame will passed into the predict_missing_value function, which simply substitute $\frac{1}{number\ of\ attributes + 1}$ into the missing values. In this case, the number of attributes is 57, excluding “filename” and “label”. Then the evaluate function will compare the training result and the actual result and output the accuracy, precision, recall and f1 score. Results are shown on (Figure 2.1). Notably, since proportions are randomly generated, every time the running result will be differently, and (figure 2.1) is a one-time result.

	proportion	accuracy	precision	recall	f1
0	0.060526	0.495	0.525323	0.495	0.473841
1	0.101013	0.480	0.507357	0.480	0.456568
2	0.104481	0.490	0.513759	0.490	0.465920
3	0.130143	0.465	0.481338	0.465	0.443429
4	0.264500	0.510	0.561339	0.510	0.496832
5	0.291451	0.470	0.504692	0.470	0.457124
6	0.438492	0.470	0.493472	0.470	0.446908
7	0.447401	0.450	0.462122	0.450	0.417498
8	0.460313	0.445	0.459052	0.445	0.416411
9	0.486867	0.415	0.440029	0.415	0.391450
10	0.525653	0.430	0.458432	0.430	0.403701
11	0.559366	0.435	0.457735	0.435	0.418482
12	0.597505	0.405	0.442453	0.405	0.382297
13	0.609836	0.410	0.427450	0.410	0.390681
14	0.628434	0.425	0.426662	0.425	0.389842
15	0.678818	0.440	0.490484	0.440	0.422747
16	0.696715	0.415	0.427017	0.415	0.391475
17	0.719538	0.395	0.411639	0.395	0.360009
18	0.849348	0.370	0.417611	0.370	0.335083
19	0.957825	0.195	0.211974	0.195	0.171969

Figure 2.1

(Figure 2.2) is a scatter plot with line of best fit based on the result of (Figure 2.1). It is clear that when the proportion of missing value going up, the accuracy/precision/recall/f1 is going down, that is the model’s performance of dealing with missing value is decreasing. Observing that when the proportion is close to 1, the evaluation factors are about 0.2, which is not good, but better than randomly guessing one attribute. Another interesting point is that only three lines are on the graph (Figure 2.2). This is because that accuracy and recall are same for this prediction, when using weighted averaging. This does not happened when using marco-averaging. The Figure 2.3 and 2.4 correspond to marco-averaging (on the page 3).

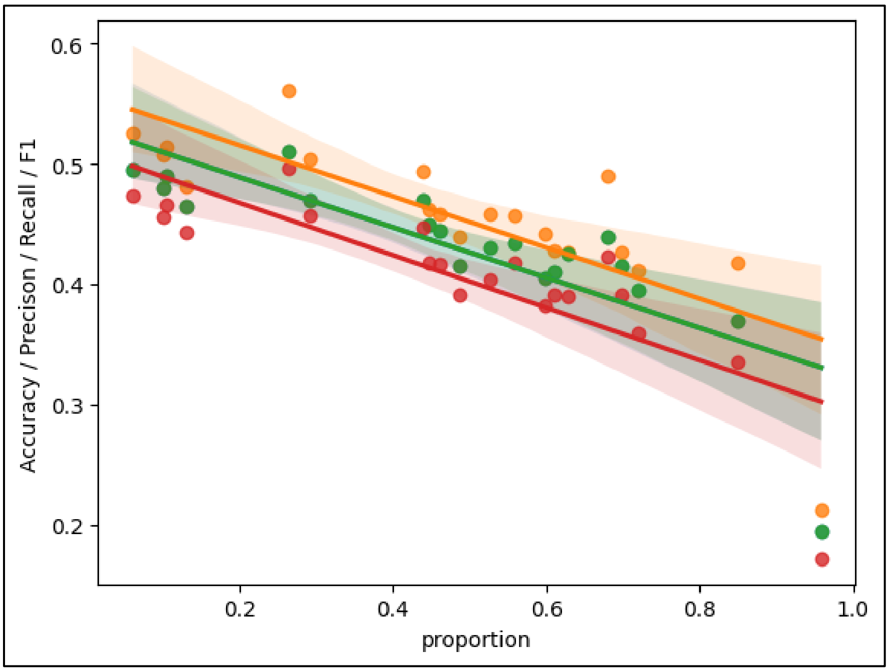

Figure 2.2

	proportion	accuracy	precision	recall	f1
0	0.002267	0.495	0.549410	0.507981	0.486664
1	0.046999	0.485	0.522587	0.494997	0.476539
2	0.144751	0.505	0.542420	0.514590	0.493161
3	0.149905	0.480	0.509471	0.489130	0.464161
4	0.221679	0.485	0.508146	0.495504	0.464921
5	0.251731	0.470	0.497502	0.475518	0.454506
6	0.279620	0.475	0.537136	0.482486	0.465325
7	0.283048	0.465	0.514138	0.470571	0.447051
8	0.386894	0.490	0.553411	0.501624	0.482691
9	0.418184	0.450	0.477260	0.457926	0.435112
10	0.436943	0.480	0.518734	0.489491	0.457444
11	0.522788	0.455	0.490747	0.458007	0.442323
12	0.553060	0.400	0.418243	0.408220	0.384105
13	0.609025	0.390	0.386362	0.394725	0.366465
14	0.704515	0.425	0.467456	0.433093	0.406916
15	0.739461	0.390	0.403824	0.393768	0.358376
16	0.810234	0.380	0.411772	0.386007	0.358365
17	0.815568	0.380	0.352557	0.384301	0.342919
18	0.890456	0.310	0.312385	0.310995	0.278856
19	0.916601	0.325	0.370676	0.336370	0.308699

Figure 2.3

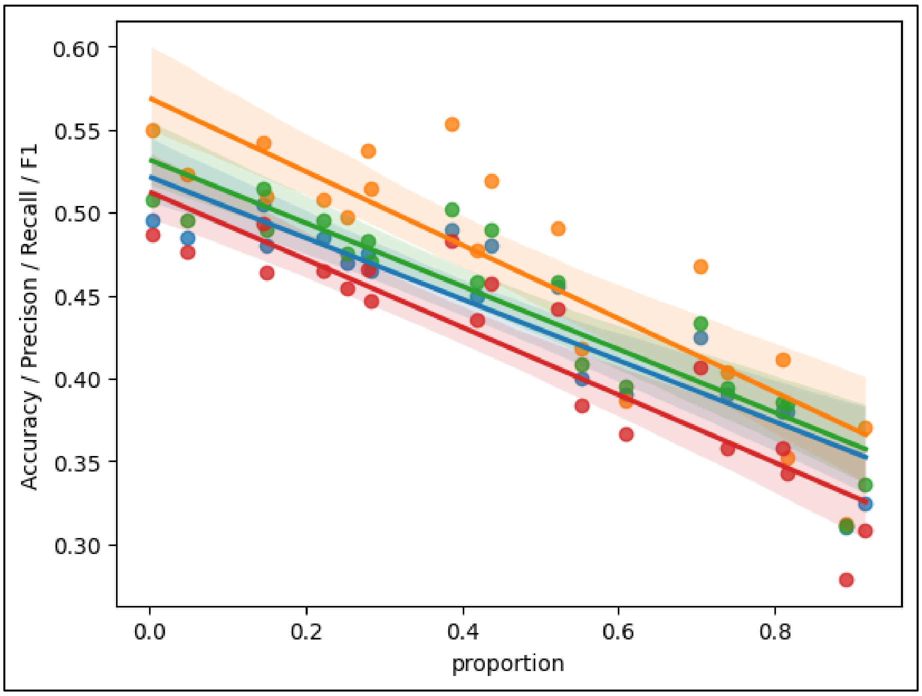

Figure 2.4