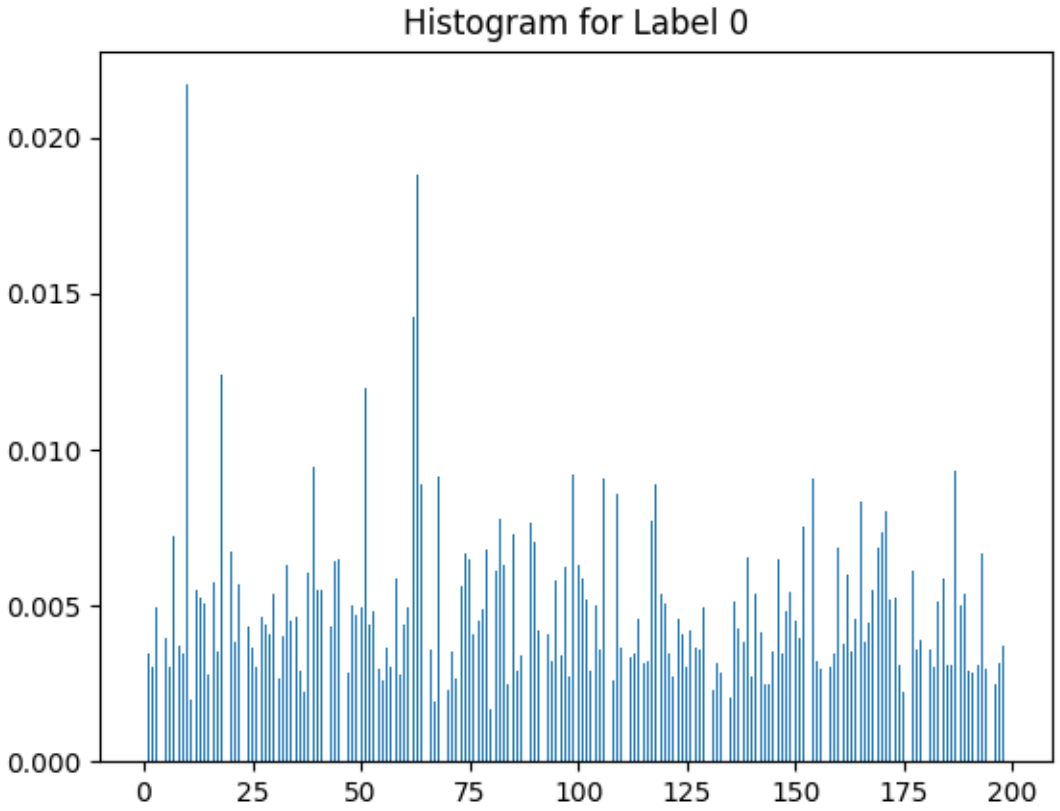
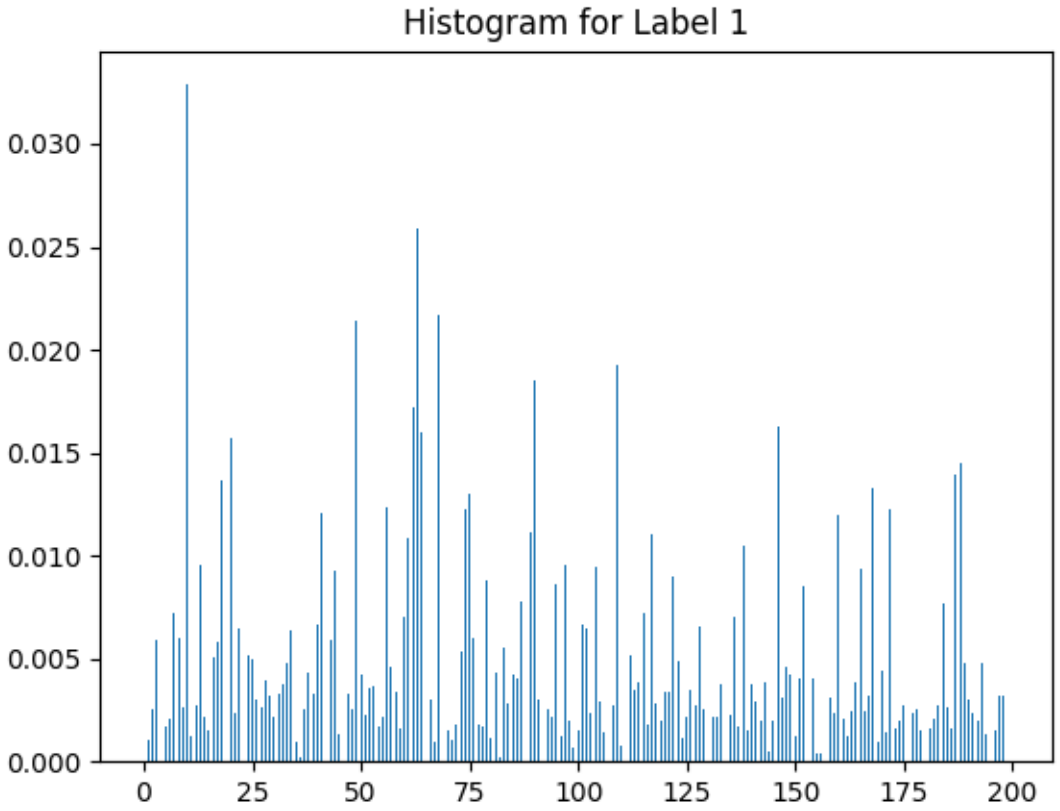
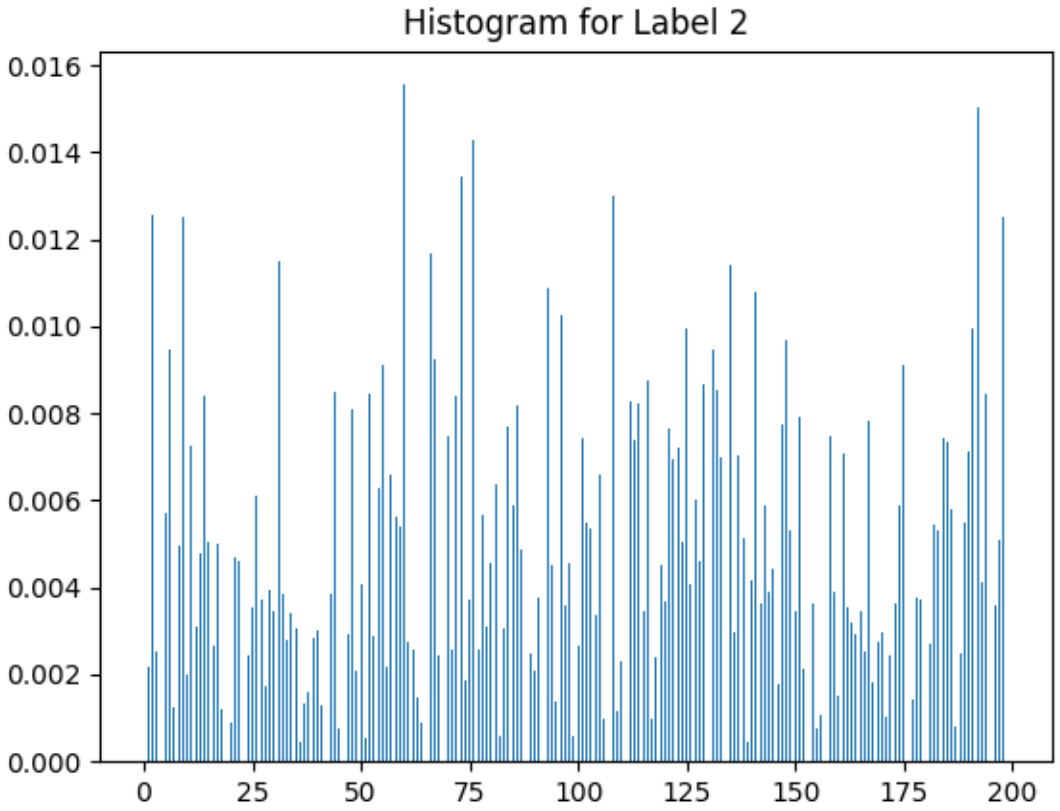
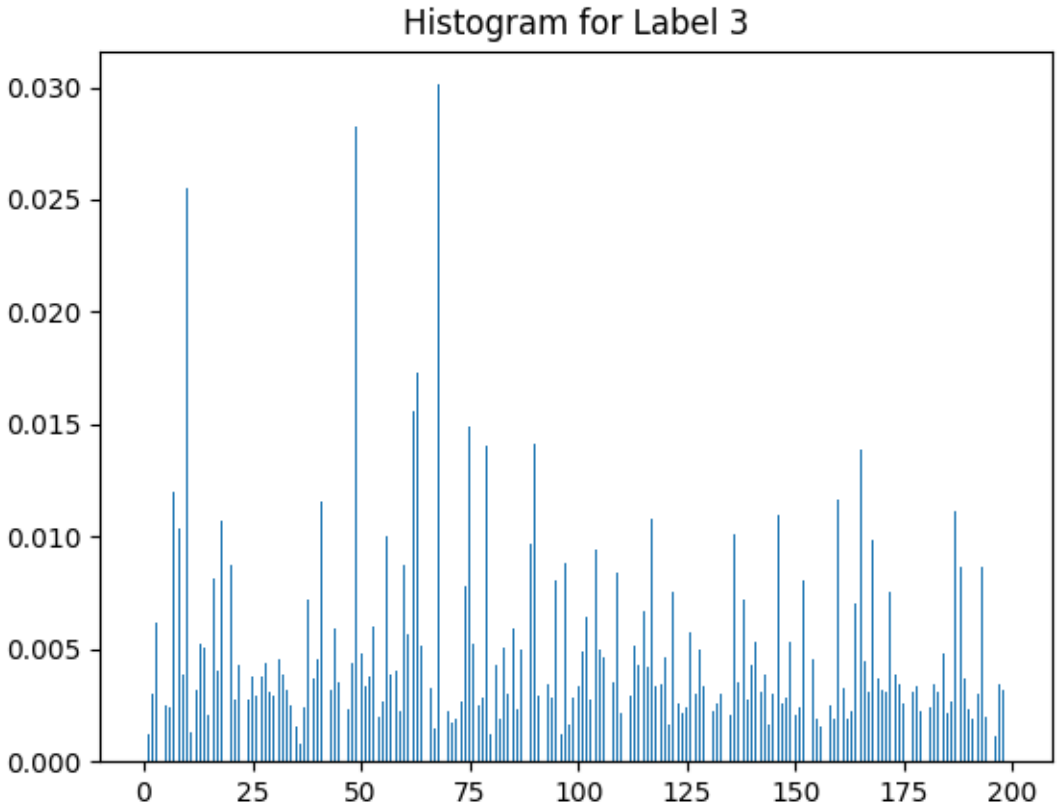
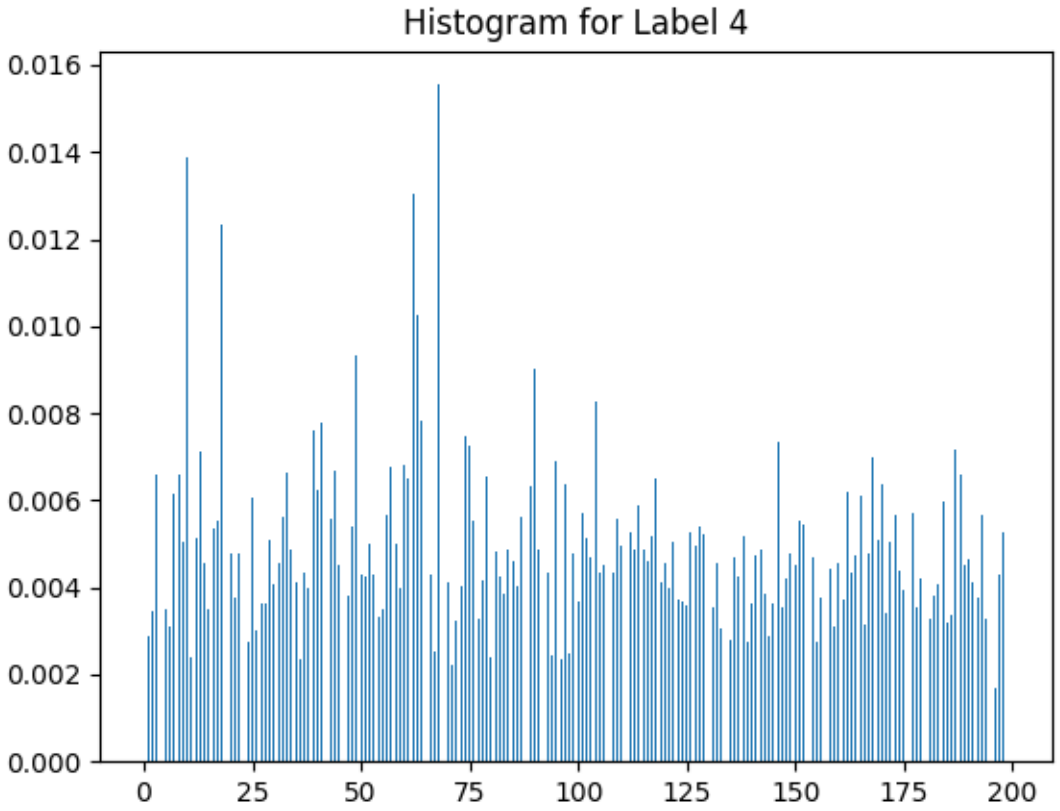
Average Histograms

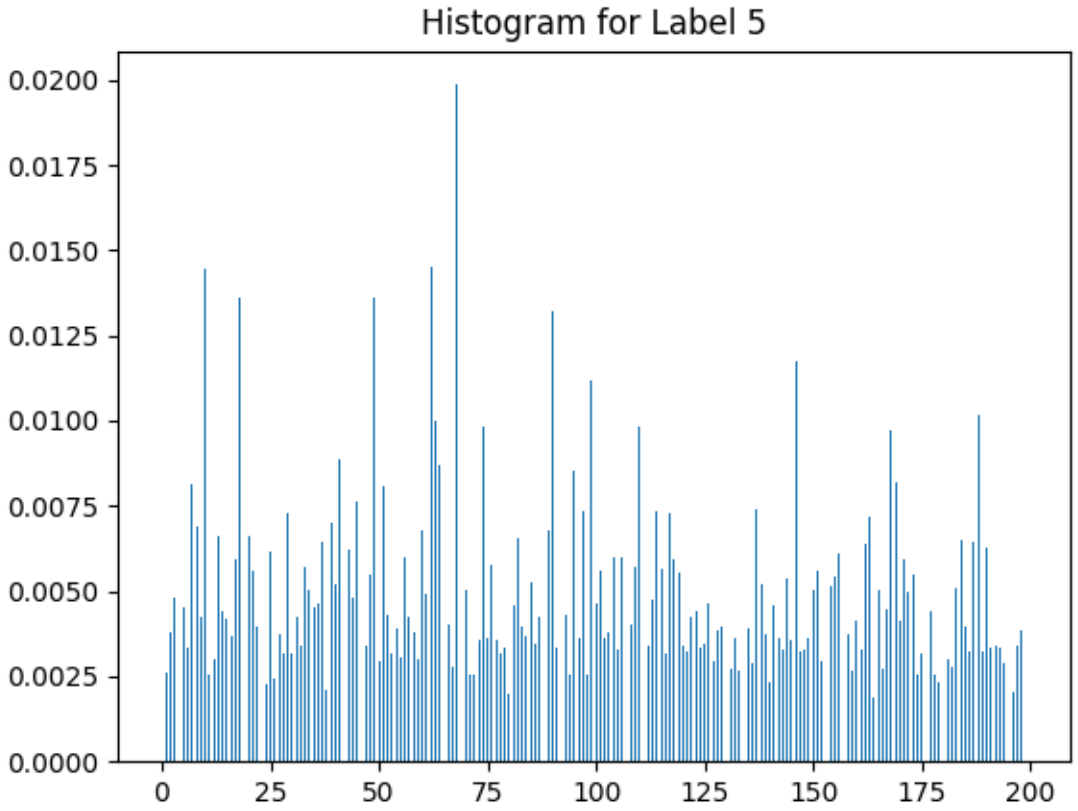


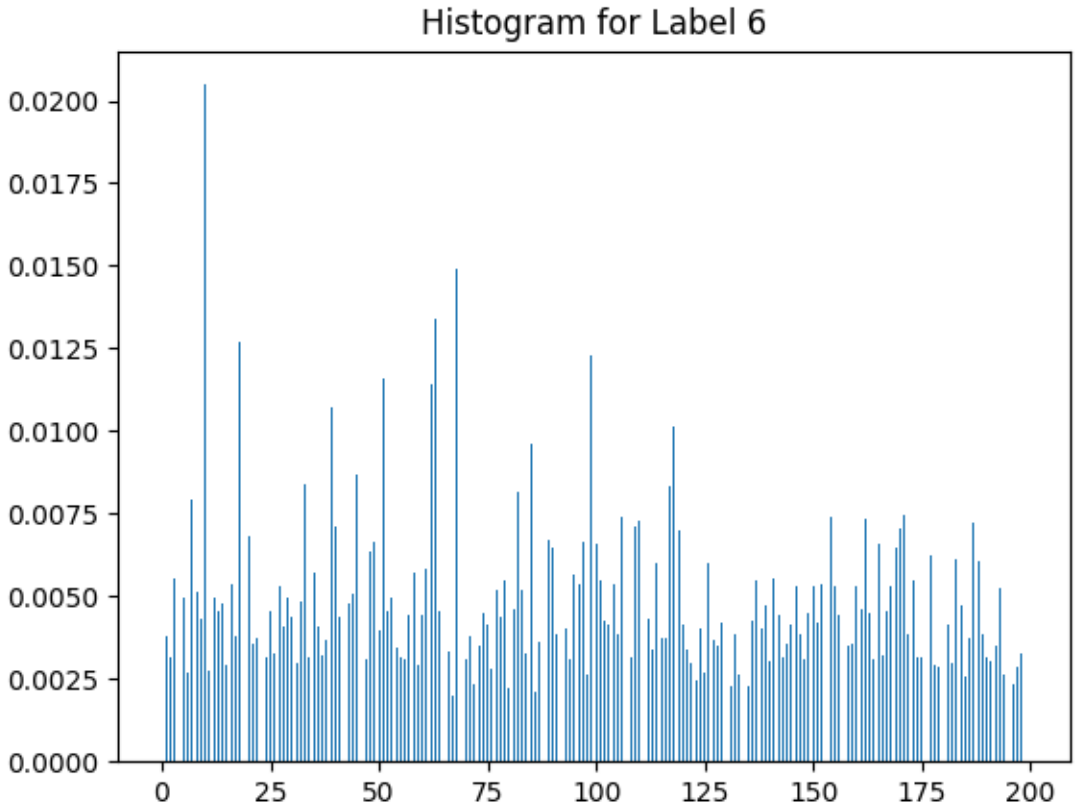

Histogram for Label 1
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
25
50
75
100
125
150
175
200

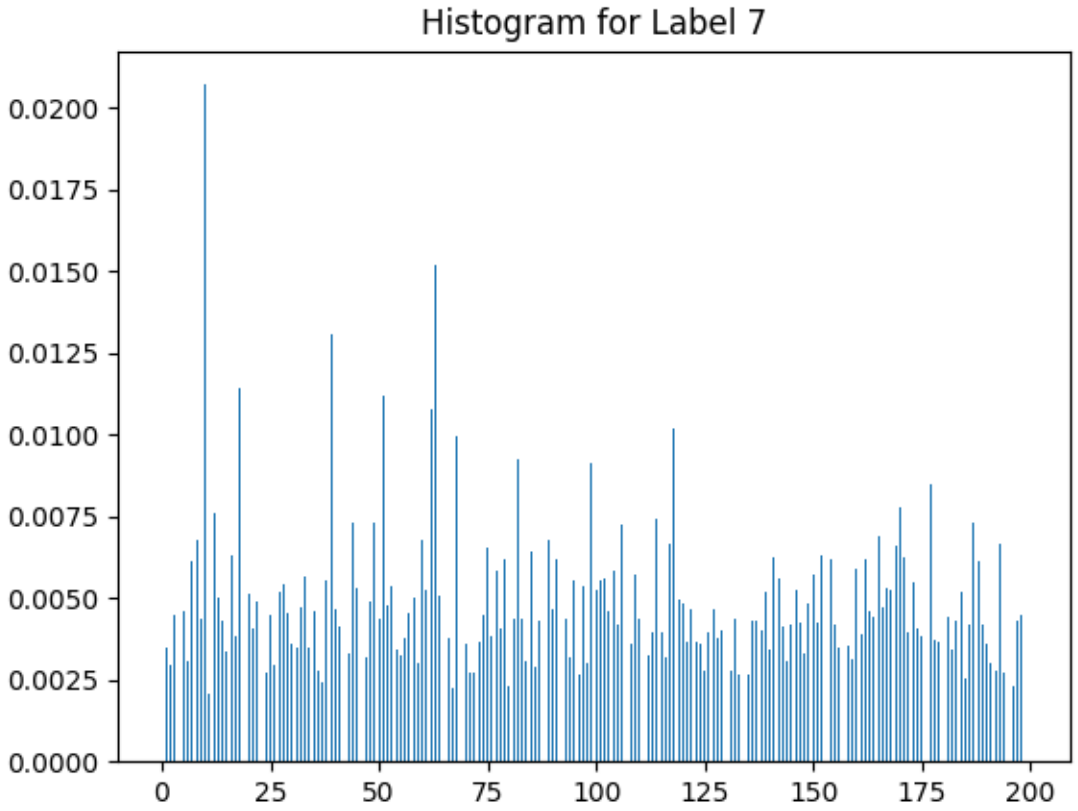

Histogram for Label 2
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
25
50
75
100
125
150
175
200

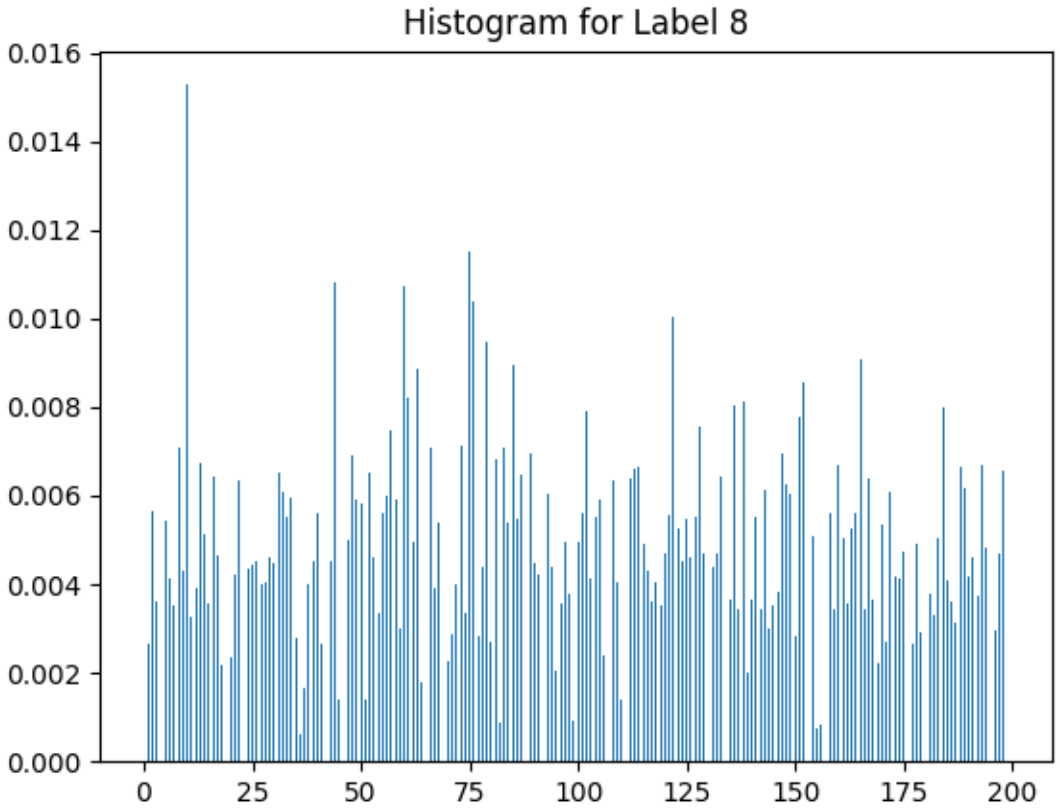

Histogram for Label 3
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
25
50
75
100
125
150
175
200

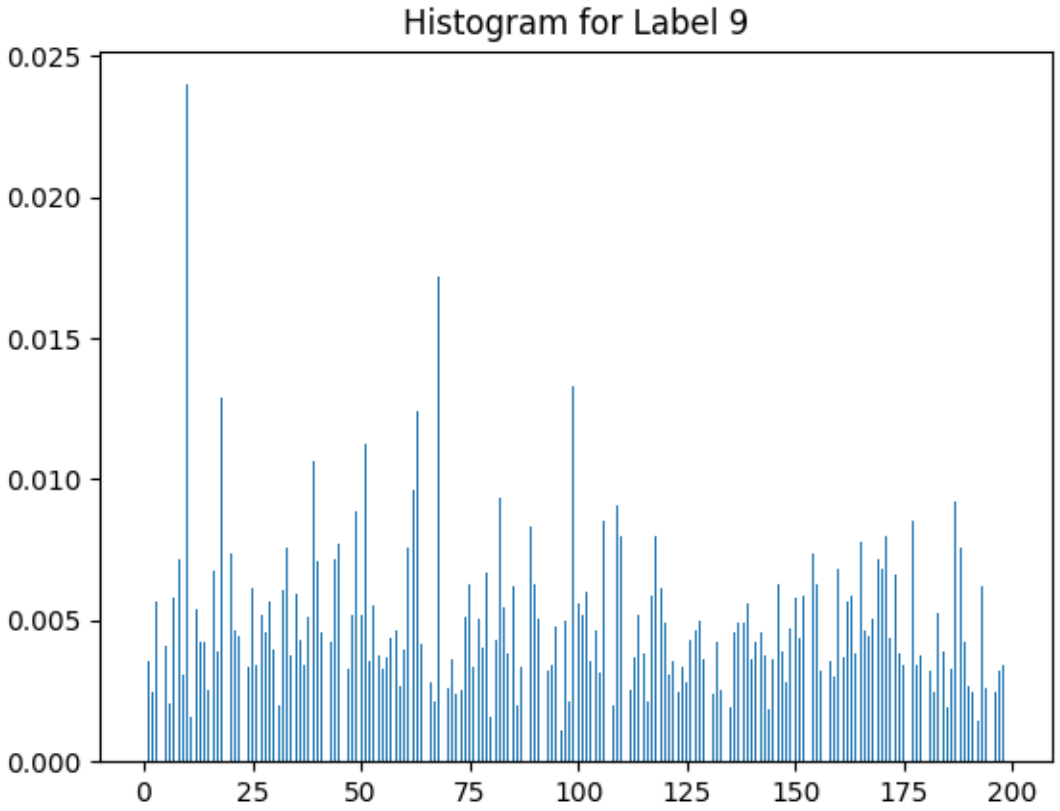

Histogram for Label 4
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
25
50
75
100
125
150
175
200

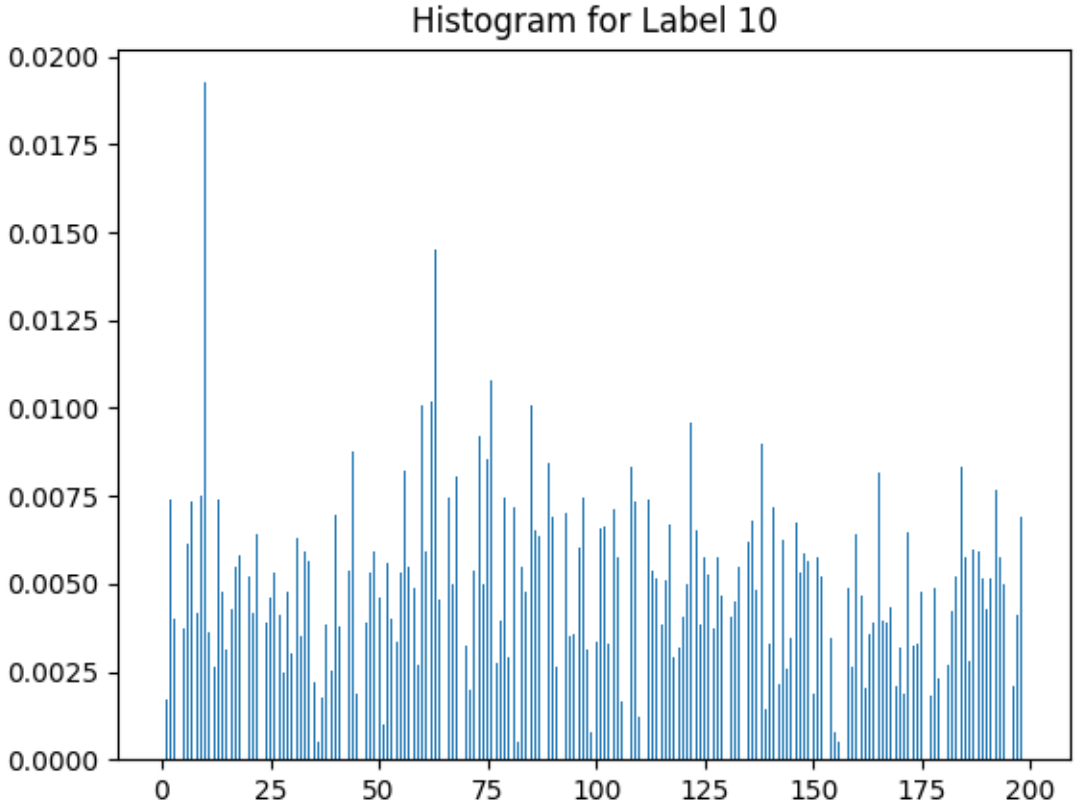

Histogram for Label 5
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
125
150
175
200

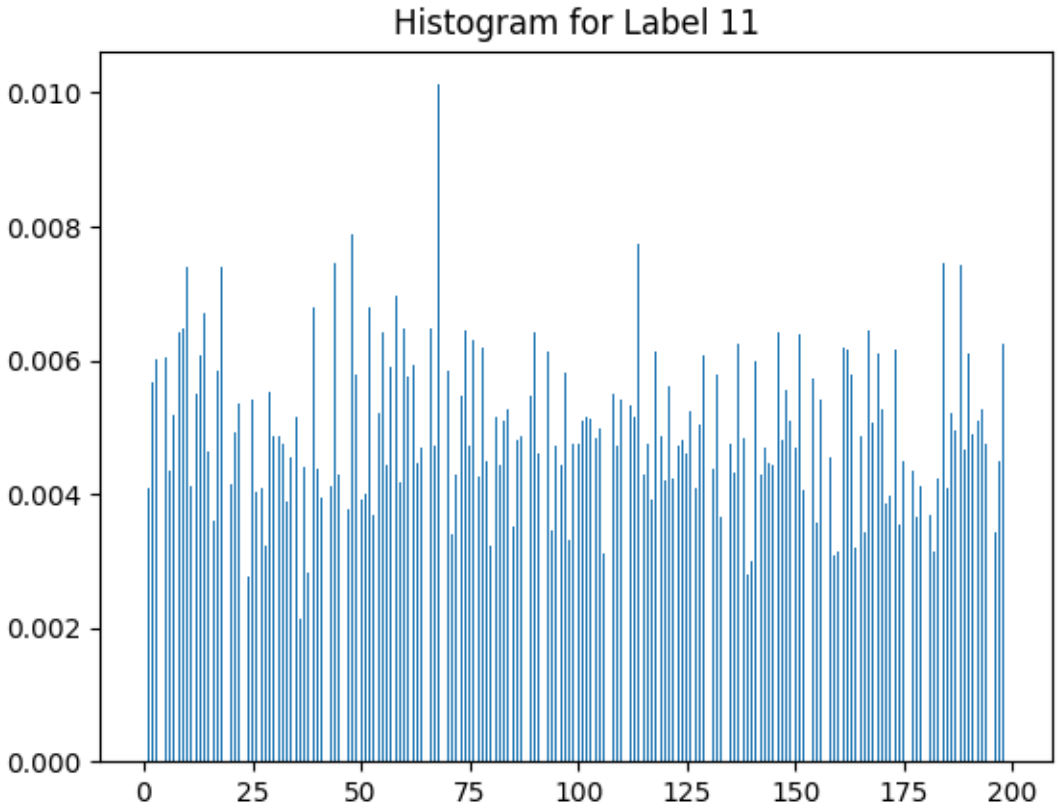

Histogram for Label 6
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
125
150
175
200

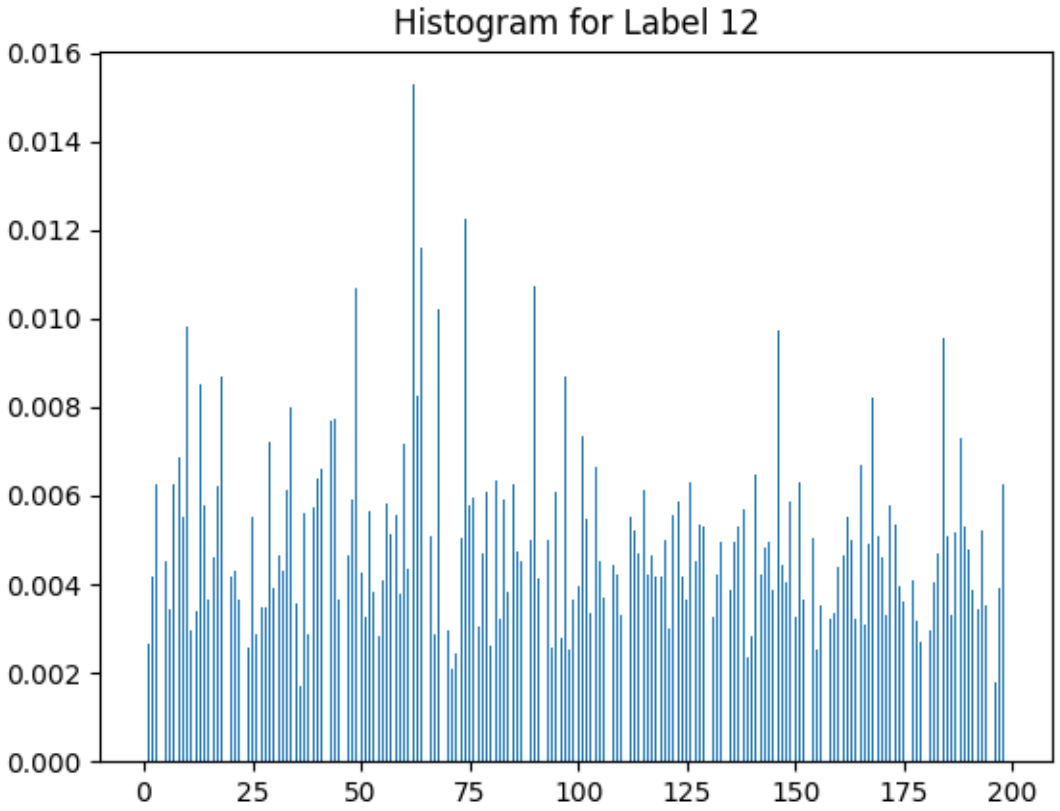

Histogram for Label 7
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
125
150
175
200

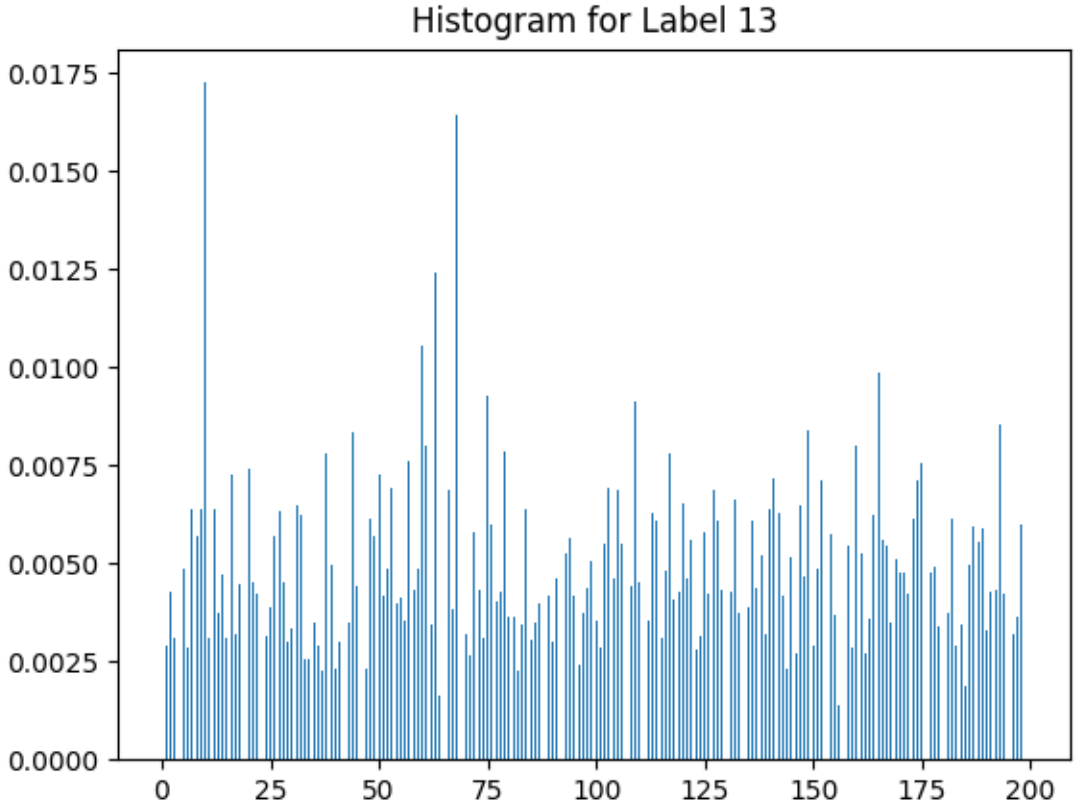

Histogram for Label 8
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
25
50
75
100
125
150
175
200

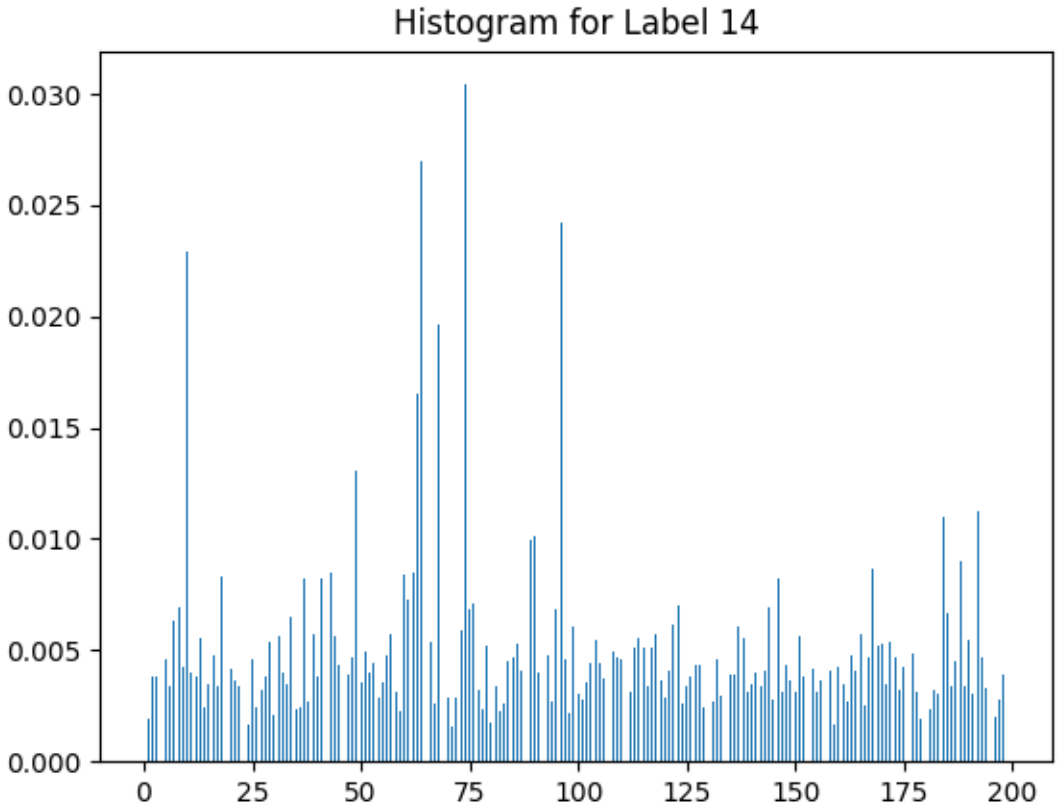

Histogram for Label 9
0.025
0.020
0.015
0.010
0.005
0.000
0
25
50
75
100
125
150
175
200


Histogram for Label 10
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
125
150
175
200


Histogram for Label 11
0.010
0.008
0.006
0.004
0.002
0.000
0
25
50
75
100
125
150
175
200


Histogram for Label 12
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
25
50
75
100
125
150
175
200

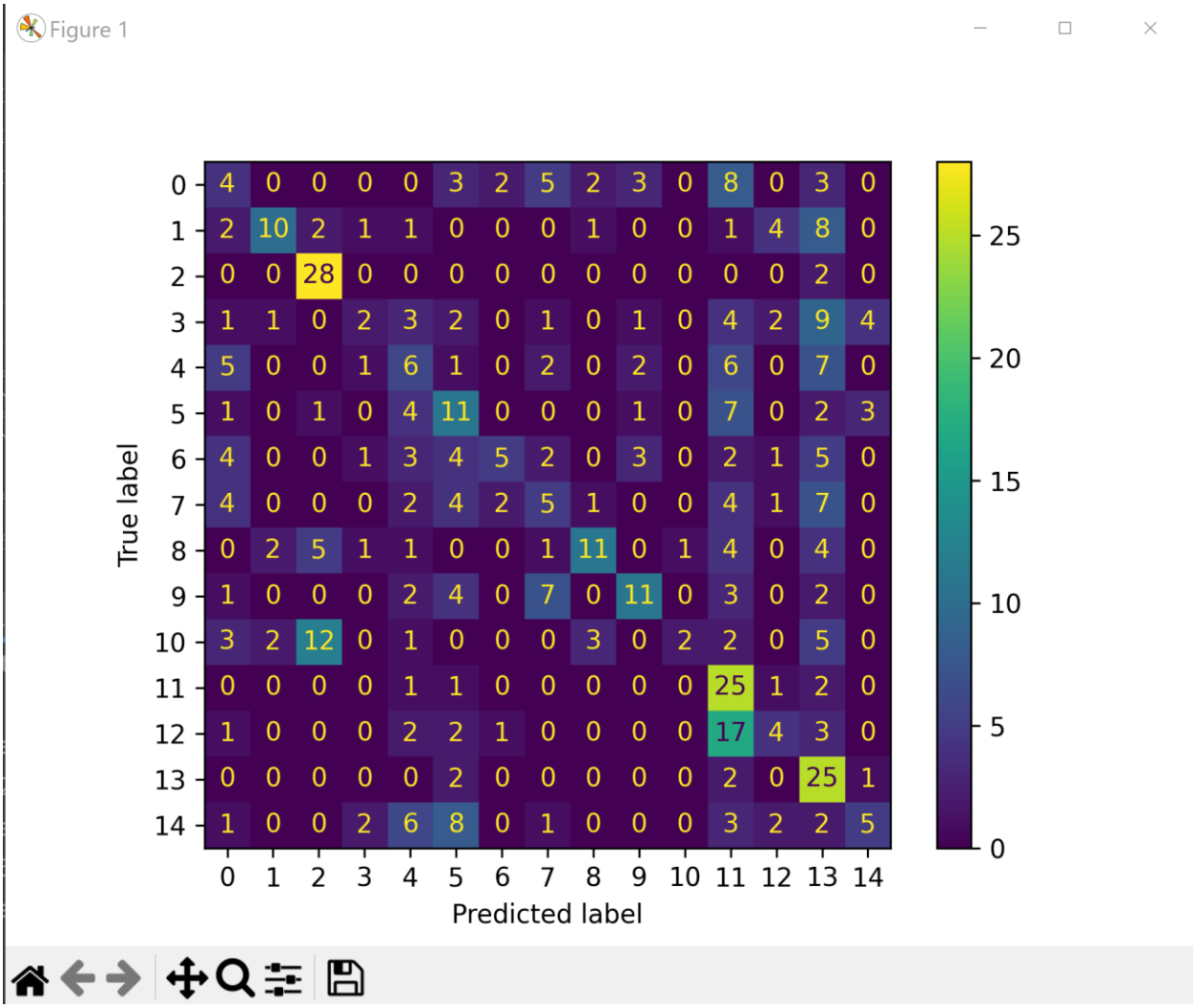

Histogram for Label 13
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
125
150
175
200



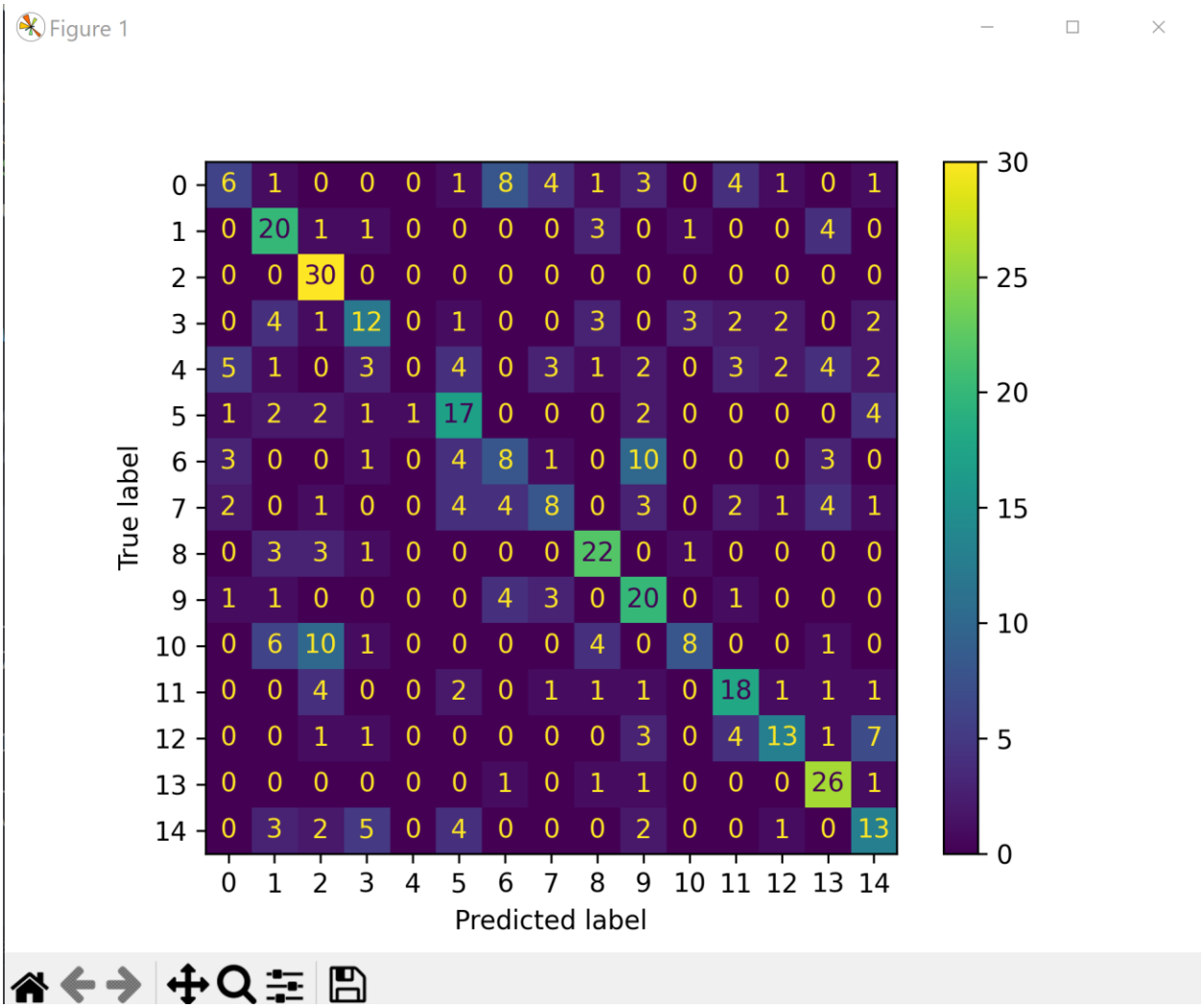
4.

As seen above, some of the histograms are much different from each other. For some of the classes it seems to do a good job at separating the data into clusters, for example class 11 where it seems to have a uniform amount across all 200 clusters. Some are not as well spread out as seen in class 14 where some clusters are far larger than the rest. I think cases like these will give worse results because it will be harder to separate them into categories.

KNN Confusion Matrix:



SVM Confusion Matrix:



KNN Accuracy Results

Run	N_neighbours	Accuracy Score
1	2	0.3022222222222222
2	5	0.3511111111111111
3	10	0.3355555555555555

SVM Accuracy Results

Run	C	Accuracy Score
1	0.1	0.3377777777777778
2	1.0	0.4266666666666667
3	5.0	0.4911111111111111
4	1000.0	0.44

Discussion

5. The size of k varied greatly depending on the value I chose. It seems that you want to pick a k value that is a happy medium between not being too big or too small. As seen in the results above, the best performing run was when I used a value in the middle. If I choose a value too small, then it will be a suboptimal score because of an overfit model. If I choose a value that is too large, then it will be suboptimal score because of an underfit model.

6. The size of C as well varied the results greatly. For C picking a value in the middle would also be important. If C is too small of a number, then the model will have too low of a penalty for misclassification which will lead to it not adjusting enough, giving a suboptimal score and underfit model. If it is too big then the model will adjust too much which will also give a suboptimal score and an overfit model. So, given this it is best to find a number in between.