CS5487 Programming Assignment 2

WANG Huanchen

October 26, 2022

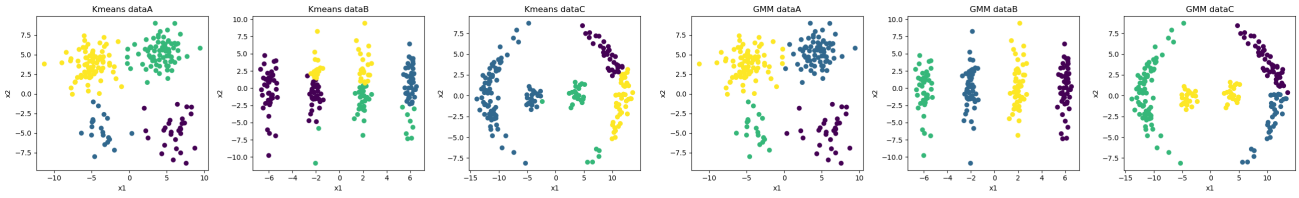
1 Problem 1: Clustering Synthetic Data

1.1 (a) Implementation

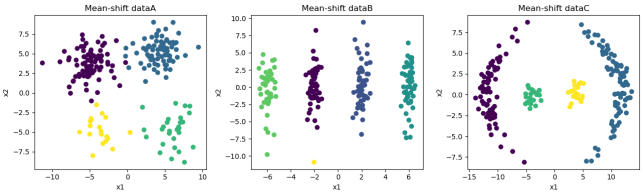
For the implementation, all the codes are in the **PA2.ipynb** file, which includes the

1) Kmeans algorithm with the scaling method for Problem 2
2) EM algorithm with Gaussian distribution
3) Mean-shift algorithm with the scaling method for Problem 2
4) Functions for loading and processing datasets and parameters
5) Functions for plotting all of the results for this Programming Assignment.
6) **External Import**
   Import the scipy.cluster.vq.whiten[1] for sample normalization and verify my scaling result in Problem 2 b), CuPy[2] for accelerating the Meanshift running in Problem 2, Pyplot[3] for plotting each result in each Problem.

1.2 (b) Running Algorithm on 3 datasets



(a) Kmeans with 3 datasets (b) GMM with 3 datasets



(c) Meanshift with 3 datasets

Figure 1: Three Algorithm

[1] https://docs.scipy.org/doc/scipy/reference/cluster.vq.html
[2] https://cupy.dev/
[3] https://matplotlib.org/3.5.1/api/_as_gen/matplotlib.pyplot.html

Kmeans The clustering results of the Kmeans algorithm are shown in Figure 1a. In data A, its result performed well, but not in data B and C. Due to the Kmeans just the case of a specific version of EM that z is a parameter rather than a hidden value and in an integer type, it cannot work well for clustering the sample with the same centroid (i.e. data b and data C). Besides, with the random centroid selection, it can also lead the non-ideal results about the clustering, even in data A, maybe we can initially select K centroids by Kmeans++.

GMM The clustering results of the EM algorithm in Gaussian distribution (GMM) are shown in Figure 1b. And it was displayed that EM performed well on data A and B, but still fail on data C. The reason for that is each sample has the responsibility in each cluster by the soft clustering. Each component has the same distribution for the selected samples, which can be useful in data A and B. But data C with the "bowl" shape can miss-cluster the sample, such as the central two parts were considered as one component. Besides, EM performed better on data B than Kmeans, but it spent more time on the E-step to calculate $\hat{z}$ and M-step for parameters calculation.

Mean-shift The clustering results of the Mean-shift algorithm are shown in Figure 1c. To guarantee the Mean-shift results, I used the h from 0.5 to 2.0 to find a good choice. After several tries at the bandwidth, I chose 1.6 as a relatively good parameter for Mean-shift. And to the results of these 3 datasets, data A and C performed well and data B performed not badly except for an outlier below the second cluster. Since the Mean-shift does not have a fixed number of clusters and it depends on the selection of bandwidth. To avoid the outlier like data b, which is caused by the redundant local maximum, I think can add a restriction with the fixed number of clusters. To the running time, Mean-shift cost much more time than the other two algorithms and it has the $O(mn^2)$ complexity.

In conclusion, the Mean-shift performed better than other algorithms in all datasets, but it also cost more running time than else.

1.3 (c) The sensitive to the h

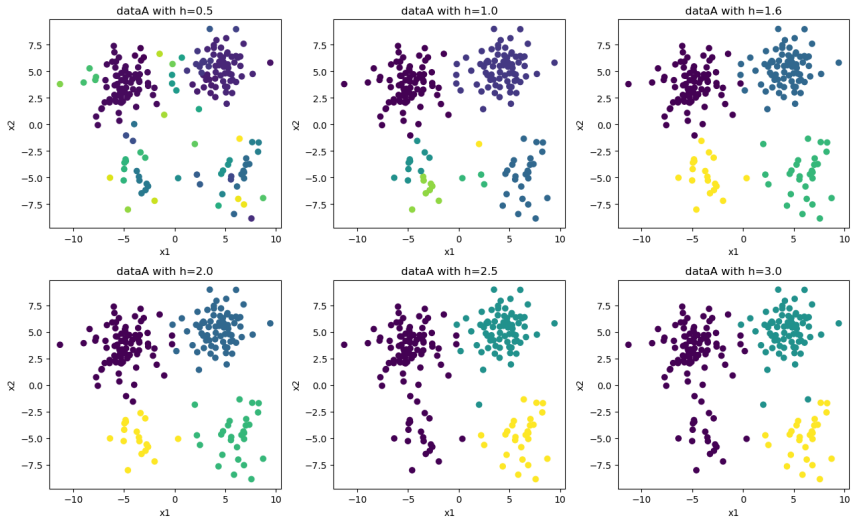
For this section, I selected 6 different bandwidth h for each dataset and plotted all of these results (Figure 2) to find the sensitivity to the bandwidth in Mean-shift.

According to these results on Figures, it indicated that Mean-shift is sensitive to the bandwidth h. What's more, the bandwidth is smaller and the number of clusters is more. In other words, the performance of the Mean-shift is based on the bandwidth selection. Besides, to these 3 datasets, we can get that h is 1.6 is more ideal than other choices. Therefore, depending on the bandwidth in the kernel function for mean shift (probable distribution), if the bandwidth is small, it can separate into a concrete approach.

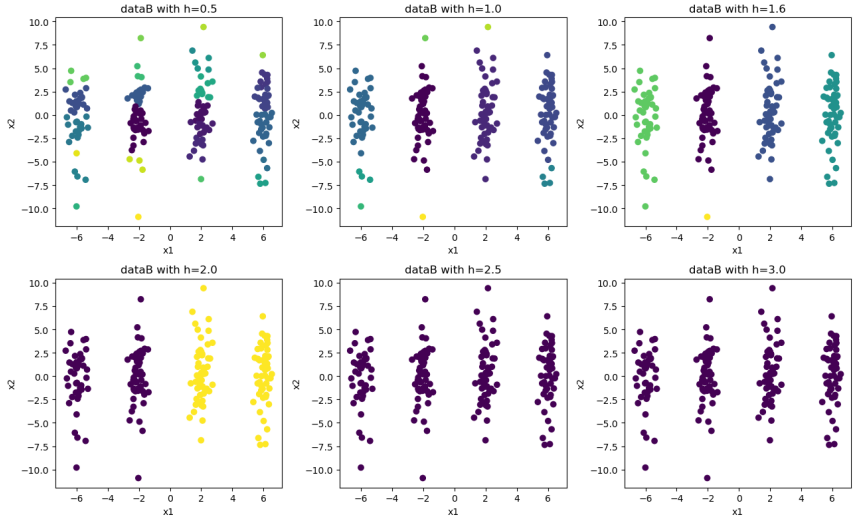
2 Problem 2: Image Segmentation

2.1 a) Segment Some Images in 3 Algorithm

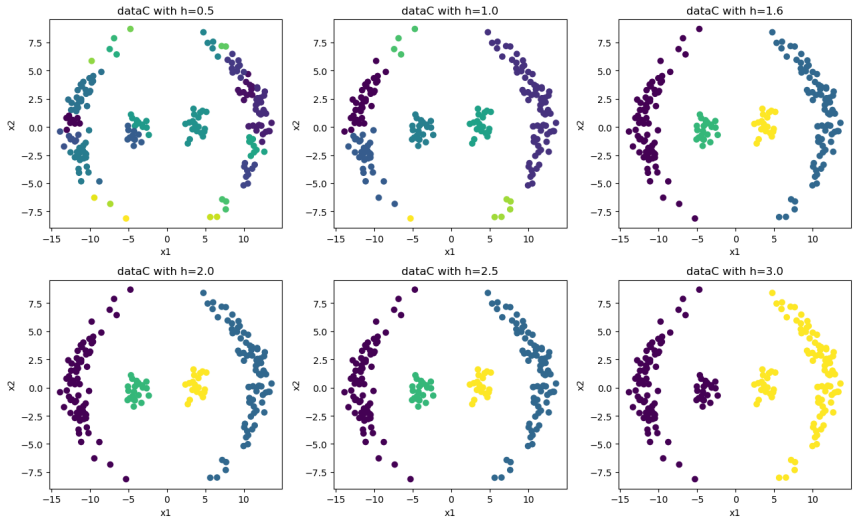
I do this work on Image 12003 and 361010. For the Kmeans and EM algorithm, I run it with the different numbers of clusters (K=2, 4, 6, 8) and also select different bandwidths in Mean-shift (h=0.2, 0.6, 1.0, 1.4) for subsequent analysis. For Image 12003, 3 algorithm results were demonstrated in Figure 3, Figure 5, Figure 7. And for Image 361010, 3 algorithm results were demonstrated in Figure 4, Figure 6, Figure 8.



(a) Meanshift on dataA with h=[0.5, 1.0, 1.6, 2.0, 2.5, 3.0]



(b) Meanshift on dataB with h=[0.5, 1.0, 1.6, 2.0, 2.5, 3.0]

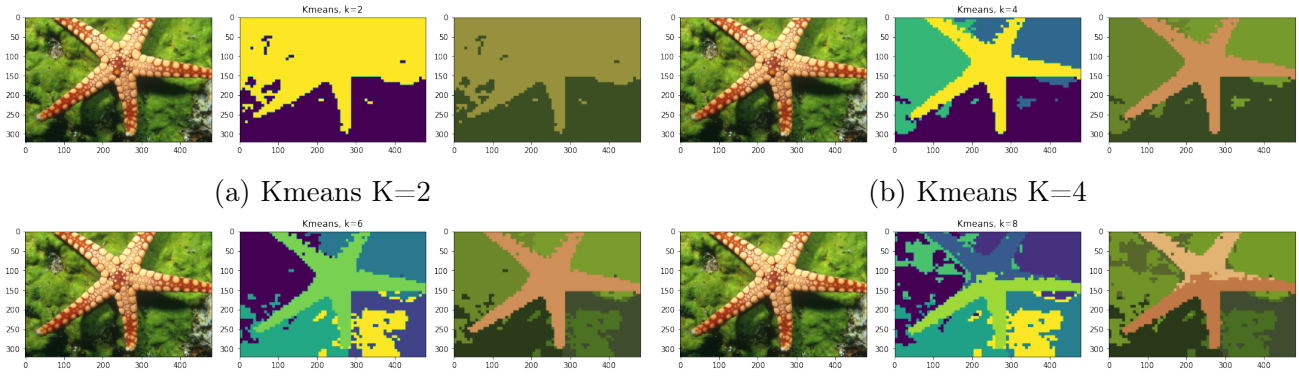


(c) Meanshift on dataC with h=[0.5, 1.0, 1.6, 2.0, 2.5, 3.0]

Figure 2: Meanshift with different h

Analysis: I select two different complexity images that Image 12003 is relatively simple (with a clear range of each object and high-contrast color) and 361010 is further hard (with ambiguous boundary and similar color) for clustering.

Kmeans For Image 12003, it revealed that when K=2, it may not perform well that cannot separate starfish and background into 2 parts with the random initial centroids selection (Figure 3a). With the K increasing, the segment displays a clear starfish but with little detail, like the grain on the starfish and background (Figure 3).

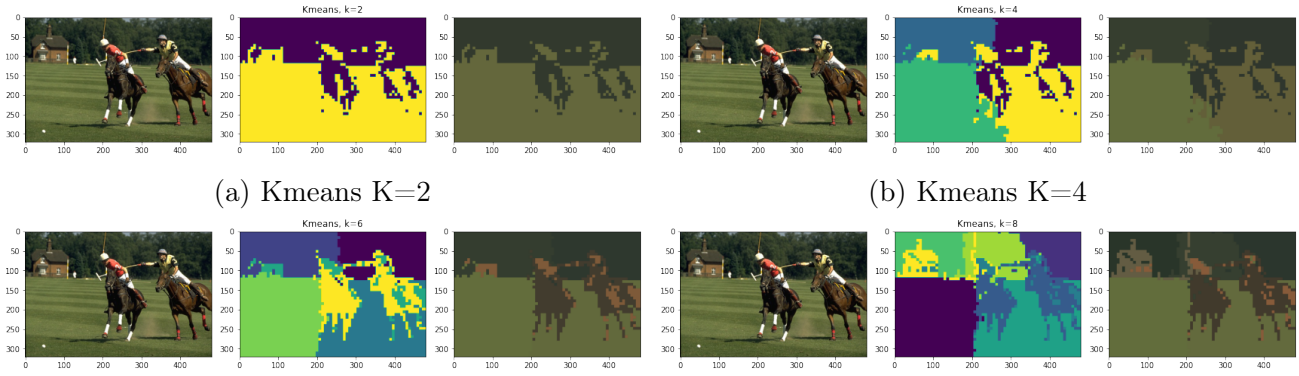


(a) Kmeans K=2 (b) Kmeans K=4

(c) Kmeans K=6 (d) Kmeans K=8

Figure 3: Kmeans for image12003

For Image 361010, it revealed interesting results in each Ks. It can roughly form the horse and athletes in digital and vague figures but they are integrated into the background (K=2), and the horse and athlete are more clear with K increasing (Figure 4a). However, Kmeans did not have a good performance in the background segment (just the coarse-grained clustering), it can just basically form the horizontal lines in the segment with colored figures (Figure 4).

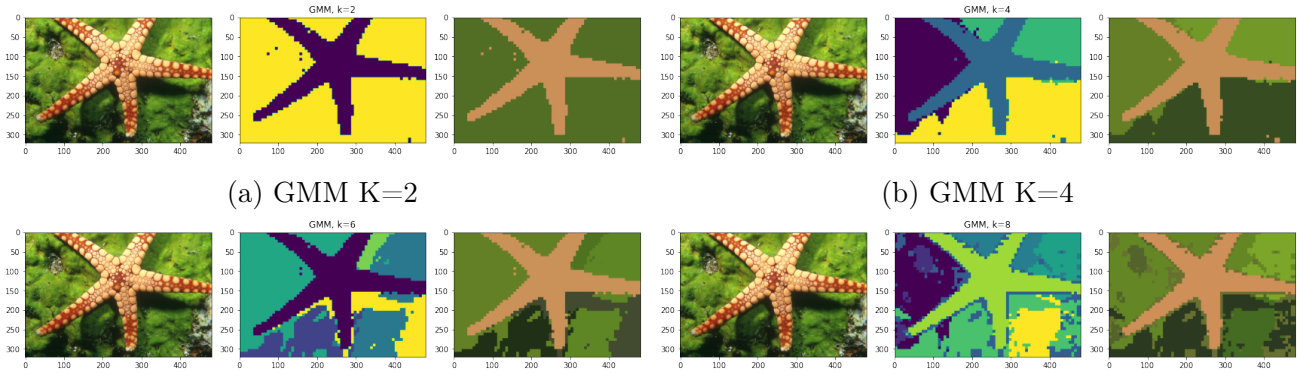


(a) Kmeans K=2 (b) Kmeans K=4

(c) Kmeans K=6 (d) Kmeans K=8

Figure 4: Kmeans for image361010

GMM For Image 12003, it revealed that all of the figures performed better than Kmeans (both K=2 can separate the starfish and background, and have detailed backgrounds with large K) in Figure 5a. Corresponding, with the K increasing, the segment displays a clear starfish and background with more detail, like the grain on the starfish and background than Kmeans results (Figure 5).

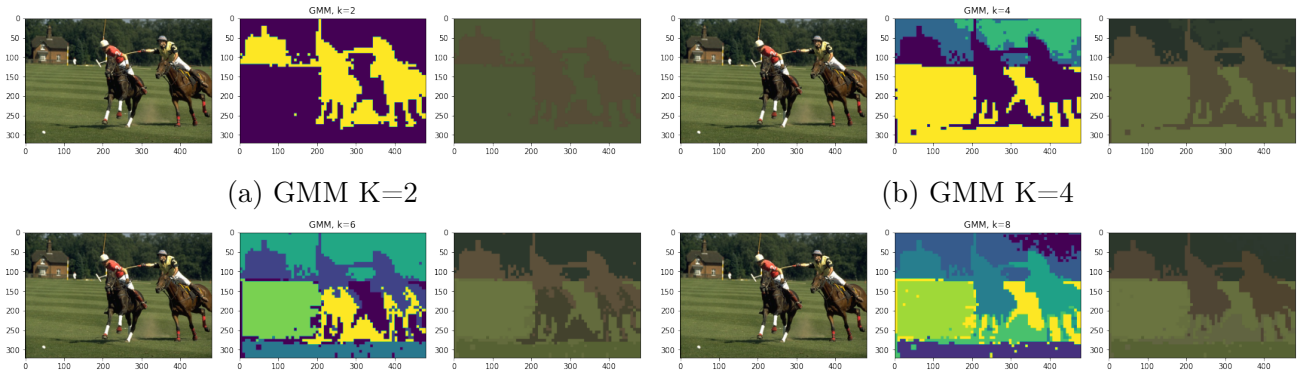


(a) GMM K=2 (b) GMM K=4

(c) GMM K=6 (d) GMM K=8

Figure 5: GMM for image12003

For Image 361010, it can easily separate the background and 3 objects (building, horses, and athletes) into 2 parts when K=2 (Figure 6a). Besides, it also displayed better performance with clear ranges of each object and playground/background, but was less grain on them, when K increased (Figure 6).



(a) GMM K=2 (b) GMM K=4

(c) GMM K=6 (d) GMM K=8

Figure 6: GMM for image361010

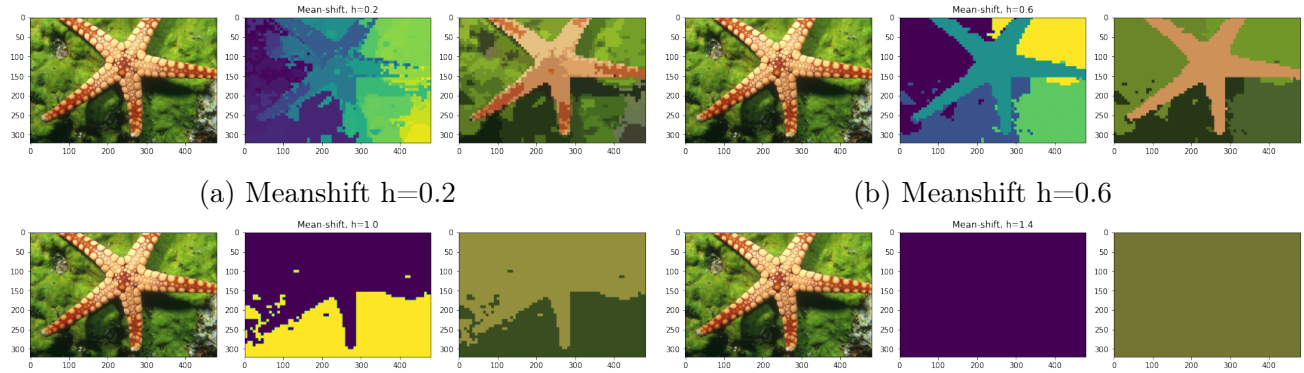
Mean-shift For Image 12003 in Mean-shift, it can easily reveal a much more detailed result than other algorithms when h=0.2 (Figure 7a). It not only has the exact boundary for each object but also many details about the grain of the starfish and background. Besides, the result can also show the Mean-sift sensitivity to the bandwidth (larger h, less detailed), even just single color like a canvas with h=1.4 (Figure 7).

For Image 361010 in Mean-shift, it showed a very interesting result like a pixel style painting when h=0.2, and all details basically included (Figure 8a). Other results were similar to Image 12003 which can nearly show a special color range, like clothes, when h=0.6 (Figure 8).

Above all, the Mean-sift needs a large running time to converge each local maximum point, but also have the best segment results when h is ideal small than the other 2 algorithms.

2.2 b) Different Feature Scaling for Kmeans and Mean-shift

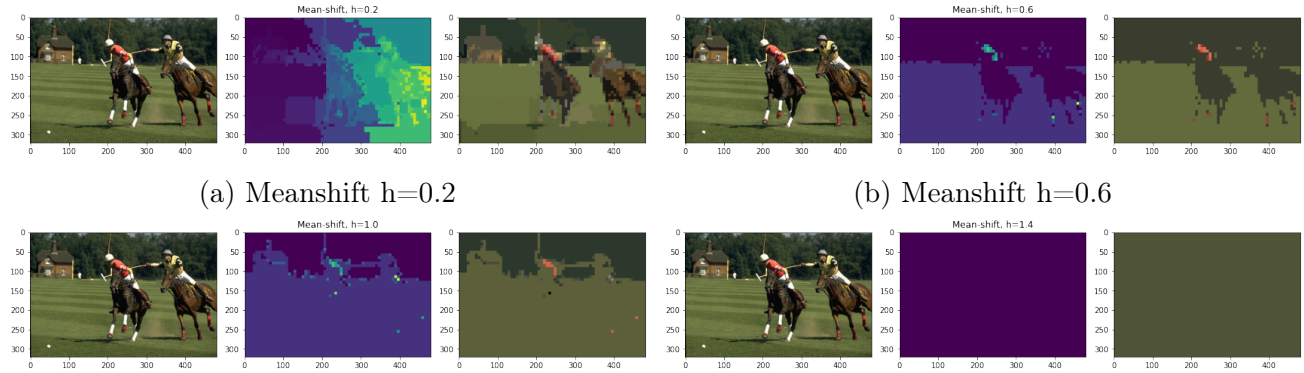
In this section, I reran the experiment with the feature scaling implementations in my codes based on the below two equations, Equation 1 is for Kmeans and Equation 2 is for Mean-shift.



(a) Meanshift h=0.2 (b) Meanshift h=0.6

(c) Meanshift h=1.0 (d) Meanshift h=1.4

Figure 7: Meanshift for image12003



(a) Meanshift h=0.2 (b) Meanshift h=0.6

(c) Meanshift h=1.0 (d) Meanshift h=1.4

Figure 8: Meanshift for image361010

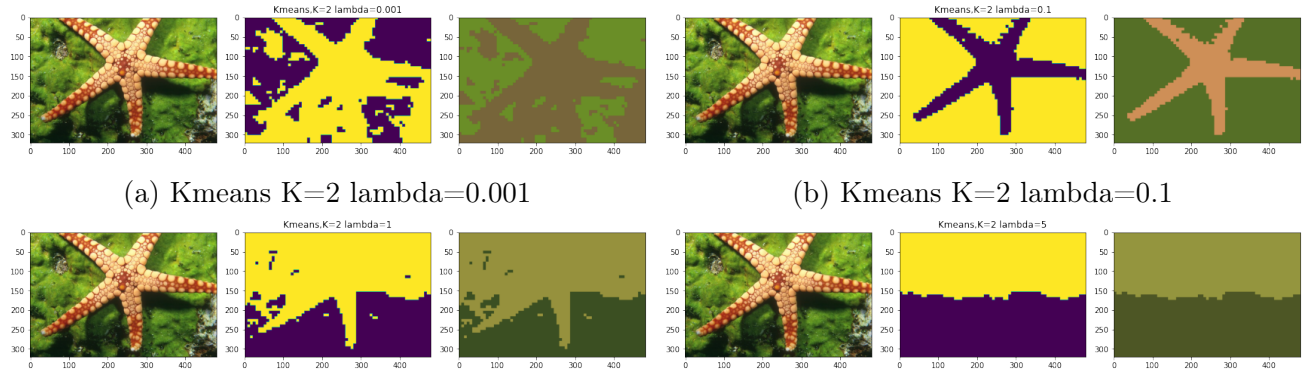
$$d\left(x, x'\right) = \left\| \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} u' \\ v' \end{bmatrix} \right\|^2 + \lambda \left\| \begin{bmatrix} c_x \\ c_y \end{bmatrix} - \begin{bmatrix} c'_x \\ c'_y \end{bmatrix} \right\|^2 \tag{1}$$

$$k\left(x, x'\right) = \frac{1}{(2\pi)^2 h_p^2 h_c^2} \exp\left\{ -\frac{1}{2h_c^2} \left\| \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} u' \\ v' \end{bmatrix} \right\|^2 - \frac{1}{2h_p^2} \left\| \begin{bmatrix} c_x \\ c_y \end{bmatrix} - \begin{bmatrix} c'_x \\ c'_y \end{bmatrix} \right\|^2 \right\} \tag{2}$$

I also selected Images 12003 and 361010 for the experiment to evaluate whether there are improvements after the feature scaling in the algorithm Kmeans add the λ and Mean-shift with h_c and h_p.

Kmeans For the Kmeans, I focus on the K=2 in Image 12003, and it can be easily found that when $\lambda = 0.1$ (Figure 9b), it can form a complete starfish boundary than previous Kmeans without feature scaling (Figure 3a). Besides, for K=4 in Image 361010, it can also demonstrate that there are much more details on 3 objects, especially when λ=0.001 and λ=0.1, than the previous result Figure 4b. What's more, the results also revealed that appropriate λ is important: too small can lead to over-clustering with some useless details (Figure 9a), and too large can lead to further rough clustering and distortion (Figure 9d).

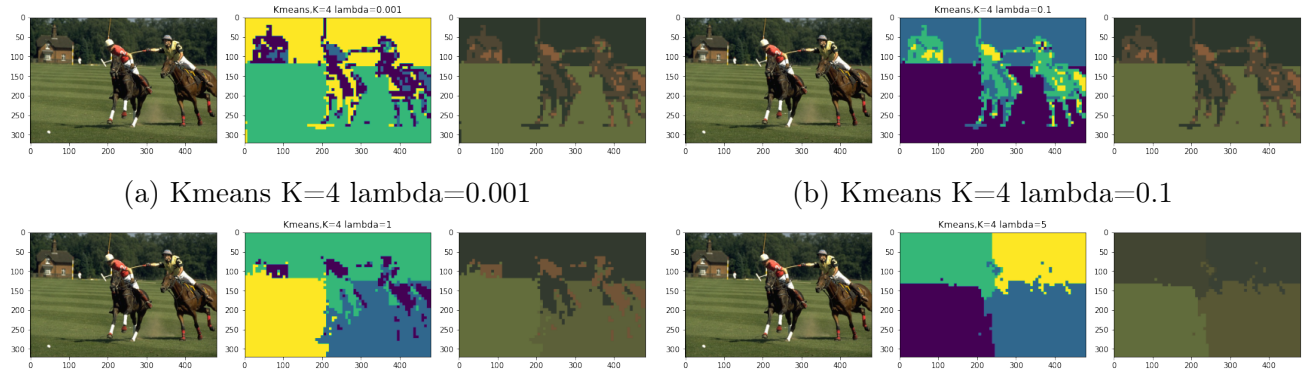
Second, for Image 361010, it had similar improvement results to Image 12003 after feature scaling Figure 10.



(a) Kmeans K=2 lambda=0.001 (b) Kmeans K=2 lambda=0.1

(c) Kmeans K=2 lambda=1 (d) Kmeans K=2 lambda=5

Figure 9: Kmeans for image12003 with features scaling

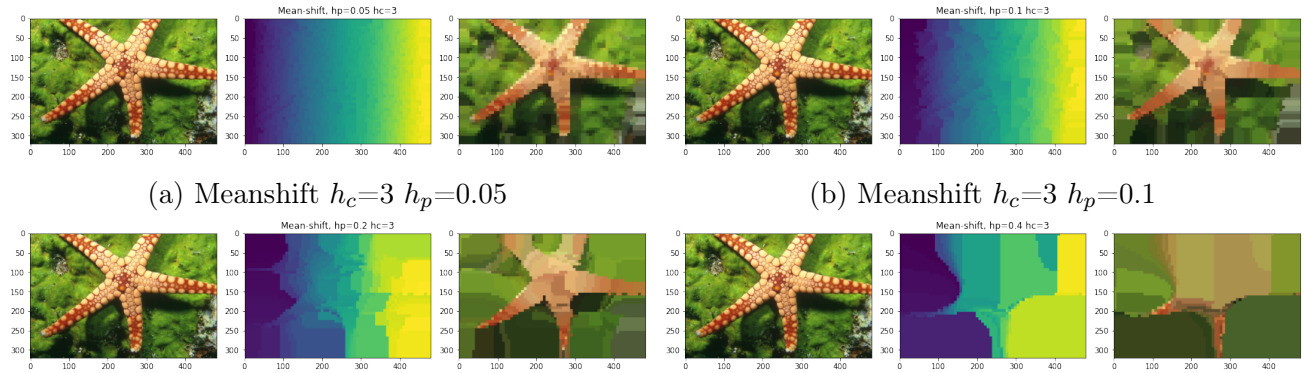


(a) Kmeans K=4 lambda=0.001 (b) Kmeans K=4 lambda=0.1

(c) Kmeans K=4 lambda=1 (d) Kmeans K=4 lambda=5

Figure 10: Kmeans for image361010 with features scaling

Mean-shift For the Mean-shift, I fixed $h_c = 3$ and firstly explored Image 12003. And it can be easily found that when h_p is smaller, it has a more realistic result even close to the original figure (Figure 11a), and it can nearly restore a comprehensive picture than the previous Mean-shift without feature scaling. In addition, the segment part in each middle plot can show the difference between scaling and not.



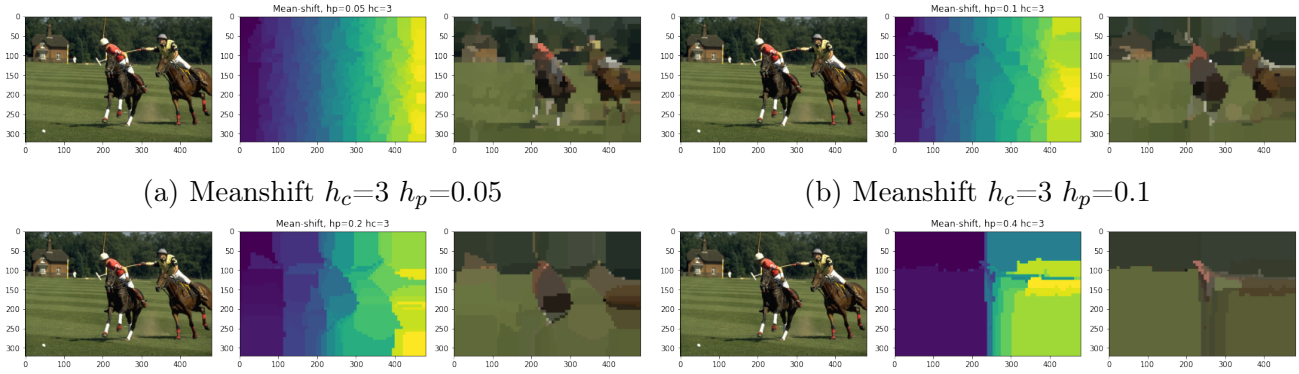
(a) Meanshift h_c=3 h_p=0.05 (b) Meanshift h_c=3 h_p=0.1

(c) Meanshift h_c=3 h_p=0.2 (d) Meanshift h_c=3 h_p=0.4

Figure 11: Meanshift for image12003 with features scaling

As Image 361010, it not only had detail on each object (i.e. horses, athletes and building)

but can display a result close to the real picture (Figure 12).



(a) Meanshift h_c=3 h_p=0.05

(b) Meanshift h_c=3 h_p=0.1

(c) Meanshift h_c=3 h_p=0.2

(d) Meanshift h_c=3 h_p=0.4

Figure 12: Meanshift for image361010 with features scaling

In conclusion, there are the segmentation results improving after the feature scaling for Kmeans and Mean-shift.