# Assignment 6 Unsupervised Learning

## Exercise 1: K-Means Clustering

**Question:** Discuss all differences and results (and their theoretical reasons) in the report. Also, think about what may cause k-means to fail (or at least underperform) and how one might reduce the risk of it happening.

The important parameter for K-Means clustering (K-Means) is the number of clusters, $k$, which is assumed to be known beforehand. With $k$ increasing, more and more clusters are generated. The parameter of iteration times is also needed to be set. In the experiment, it is set to be 100 while the algorithm always converges at no maximum than 20 iterations. Therefore, it does not influence on the final results.

One obvious problem of K-Means is that it can only reach the local minimum of the initially randomized centers while not the global minimum. Therefore, it is quite sensitive to the locations of the initial centers, which in our case is randomly chosen without any restrictions. The above point can be illustrated in the following figure.

As easily observed from the fixed data distribution, there should be six clusters classified as Figure 1(a) while this is not always the case. The final results could also be in Figure 1(b), 1(c), and 1(d) because the initially randomized centers are not given properly. To improve it, some restrictions could be added to the initialization. For example, one distance threshold can be set to let the initialized centers not distribute too close spatially.

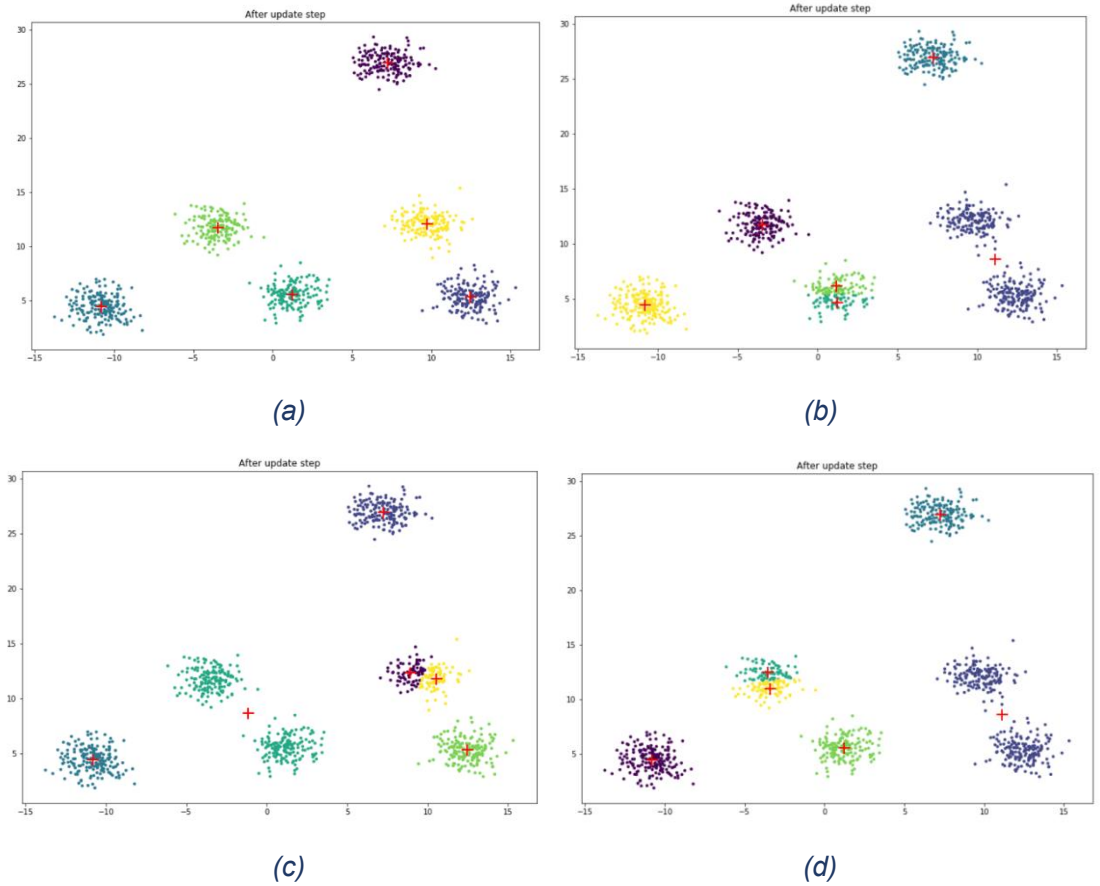


*Figure 1. Results of the fixed data with $k = 6$ by K-Means.*

For the fixed data distribution above, we also tried the case with $k = 2$ and $k = 10$ as shown in Figure 2. As we can see, the results for fixed data distribution are not as good as the case with $k = 6$. With the same $k$ setting, the results are not optimal for different data distribution as in Figure 3, whose data is randomly generated.

In conclusion, there exists a most meaningful number of clusters for different datasets. If there isn't previously known information, to find the optimal k, the metrics like intra-class variance and between-class variance can be computed for different k and plot them by two curves. The optimal one with small intra-class variance and large between-class variance should be found near the knee points of the curves.

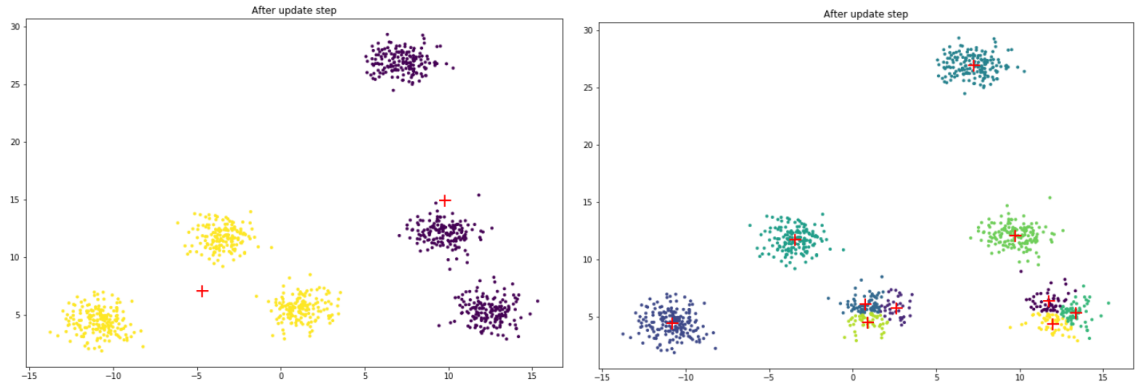


*Figure 2. Results of the fixed data with $k = 2$ (left) and $k = 10$ (right) by K-Means.*

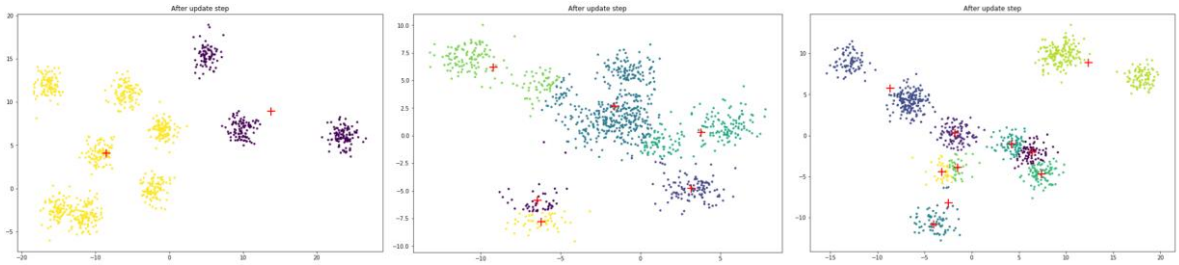


*Figure 3. Results of the random data with $k = 2$ (left), $k = 6$ (left) and $k = 10$ (right) by K-Means.*

## Exercise 2: Kernel density estimation

**Question:** Compare outcomes of the different kernels for varying bandwidths in your report. Last, run data with gaps. Again, compare all results in your report and explain reasons for performance differences (between kernels, bandwidths, and compared to complete data samples of 1d) based on the theory.

Rather than giving a step function for each sample point, kernel density estimation (KDE) gives more weights to data that are closer to the query points. Two most important parameters appear here, the bandwidth $h$ in which data points are considered to be close to assign weights, and the weight function.

Intuitively, one wants to choose the bandwidth as small as enough to be as accurate as enough at that point. While the points could be noises and generates an unsmooth curve with too many spurious data artifacts (high variance). With too large bandwidth, a wider range of points is considered and thus generate an over-smoothed curve with some underlying structures obscured. Therefore, it is quite crucial to choose an appropriate bandwidth.

As shown in Figure 4, both Epanechnikov kernel with $h = 0.5$ and Parzen kernel with $h = 1$ work very well. They are almost overlapped with the true data distribution. To conclude, with suitable bandwidth, several types of kernel functions might fit very well so that the choice of the bandwidth is more important than the kernel function. For the normal kernel, the results are much worse when bandwidth is 1 or 2. The single Gaussian model here does not fit to a bimodal distribution since it infinitely supports the surrounding data and thus considers the data from the other peak acting as noises. While it is much more meaningful with bandwidth as 0.5 since it decreases weights for the noise data far away from it from the other peak.

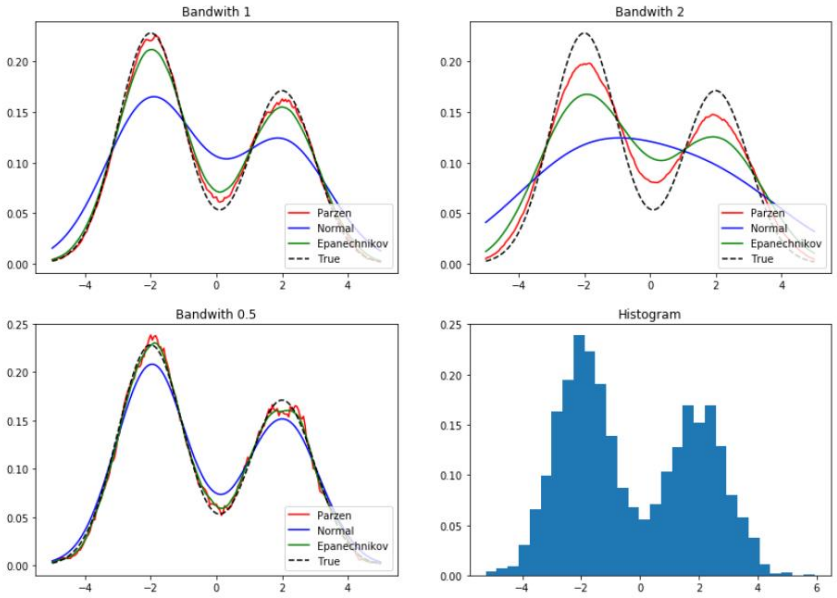


*Figure 4. Results of complete data with different kernels and bandwidth by KDE.*

To compare the complete datasets and partial ones in one dimension, the Epanechnikov kernel with $h = 1$ is still effective. The normal kernel with $h = 0.5$ works also well. However, the disadvantage of the Parzen kernel appears quite obviously here with not complete data, that there are many jumps at the boundary. By setting larger bandwidth as 2, the curve gets better while still not smooth enough compared to others. The results are shown in Figure 5. The 2d random data results are shown in Figure 6.

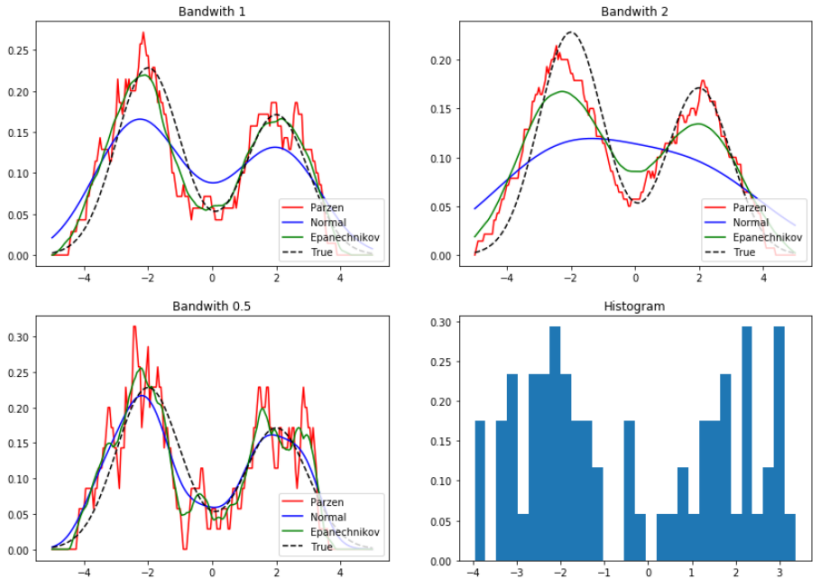


*Figure 5. Results of partial data with different kernels and bandwidth by KDE.*

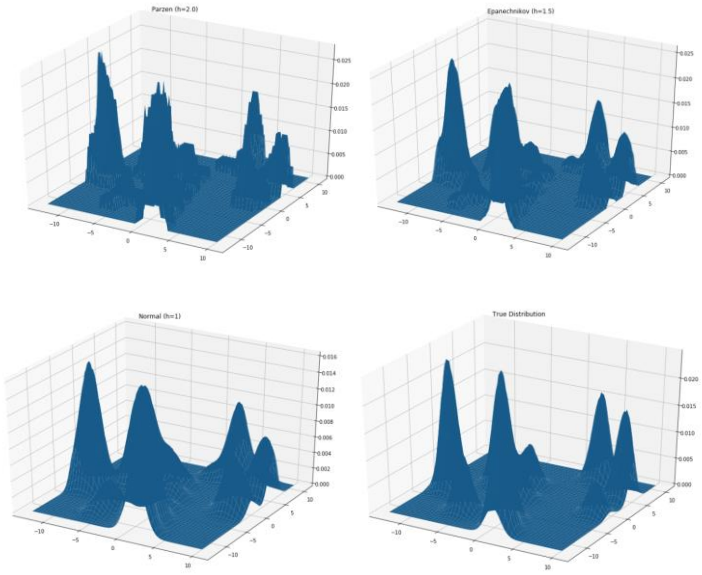

*Figure 6. Results of 2d random data with different kernels by Mean-shift.*

## Exercise 3: Mean-shift algorithm

**Question:** Compare the results of k-means and mean-shift and explain performance differences based on the theory behind both algorithms. Finally, run sampled data points randomly and again compare both k-means and mean-shift in your report. Also, try multiple times for fixed mean-shift parameter settings and report your findings.

The mean-shift algorithm (Mean-shift) gives more accurate results than the K-Means since it decides the number of clusters concerning the data distribution. By iteratively finding the high-density regions, the mean shift algorithm shifts the window's center with the bandwidth parameter to that location, until convergence is reached. The only parameter is the bandwidth and it has a real physical meaning, i.e., to what extent surrounding points are to be considered, which is unlike the number of clusters in K-Means. Also, the output of the mean shift is not dependent on the initialized center.

As shown in Figure 6, with the same dataset, unlike Figure 1's results which are largely affected by initialized centers, and Figure 2's when the optimal number of clusters assumed to be known to get a good result, Mean-shift has a much better and stable clustering result. This can also be proved by random datasets in Figure 7. No matter how the dataset is given, Mean-shift can always give a reasonable clustering result, which K-Means cannot do.

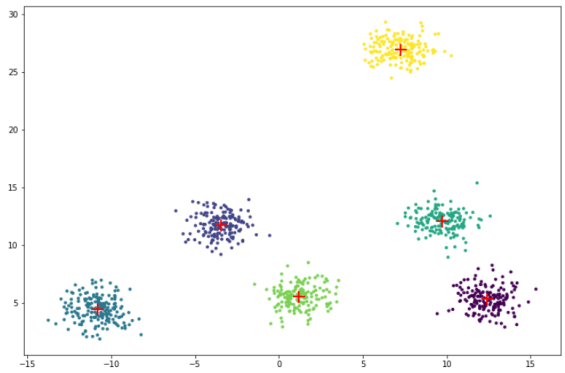

*Figure 7. Results of the fixed data by Mean-shift.*

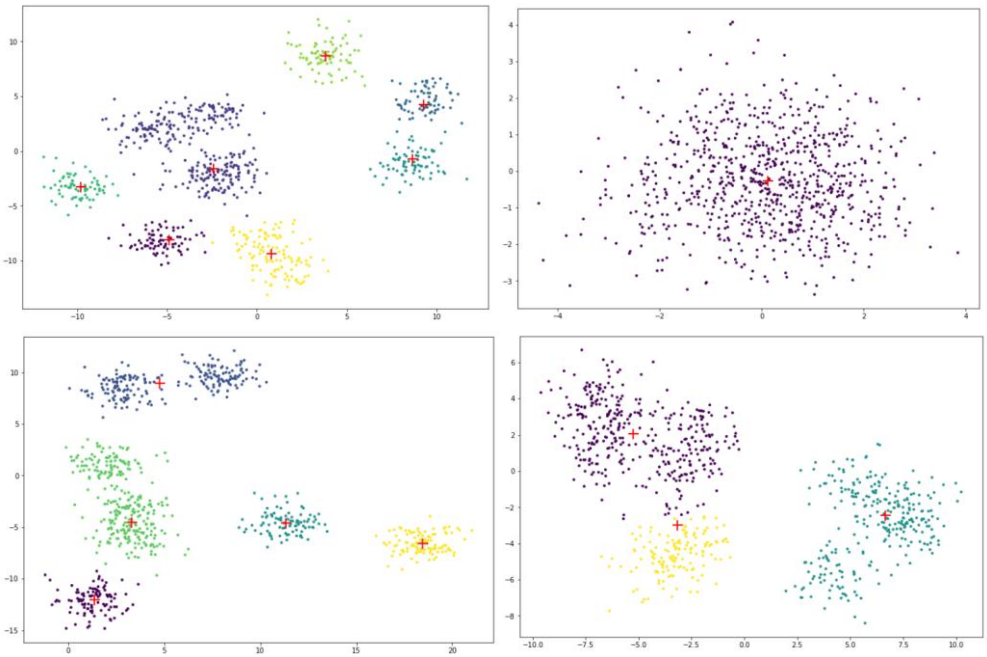

*Figure 8. Results of the random data by Mean-shift.*

To test the influence of the bandwidth on the Mean-shift algorithm, bandwidth as 1 and as 9 was tried and plotted as in Figure 8. It can be easily observed, with a very small bandwidth, there are too many unreasonable clusters since it considers only a few surrounding points, and vice versa for a very large bandwidth some information is lost and thus only partial clusters are detected.

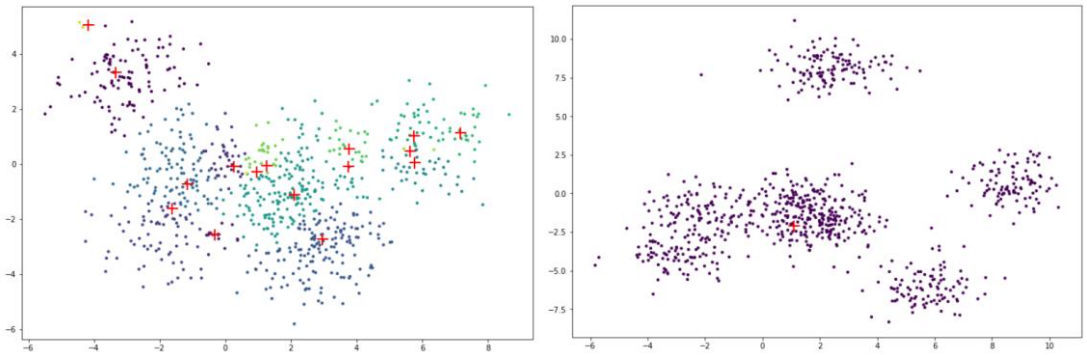

*Figure 9. Results of the random data with bandwidth as 1 (left) and as 9 (right) by Mean-shift.*