Foundations of Machine Learning Assessed Work

RUN YING JIANG (rj1u20@soton.ac.uk)

I. A

The completion of the previous five labs has been confirmed and I have corrected the Lab One of the mistake on numerical accuracy.

II. B

In this lab, K-Means Clustering was chosen to be submitted. The aim of the question was to implement and study some aspects of the K-means clustering algorithm.

A. Result of my K-means iteration

In this section, I sampled data from a mixture Gaussian density with the means of $\begin{pmatrix}0\\3\end{pmatrix}$, $\begin{pmatrix}3\\0\end{pmatrix}$, $\begin{pmatrix}4\\4\end{pmatrix}$ and covariances of a random symmetric, positive-define 2-D matrix. And I implemented K-means iterations with my own code:

```
def k_means_implement(K, X_data):
    mean_init = np.array(random_init_list(
    X_data.min(), X_data.max(), K))
    predict_init = pd.DataFrame(data=np.random.randint(
    K, size=len(X_data)), columns=['label'])
    updated_predict = predict_init
    J = 0
    while True:
        updated_mean = update_means(
        K, X_data, updated_predict)
        updated_predict = pd.DataFrame(
        data=update_predict(X_data, updated_mean),
        columns=['label'])
        J_previous = J
        J = compute_distortion_measure(
        K, X_data, updated_predict, updated_mean)
        if J == J_previous:
            break
    return J, updated_mean, updated_predict
```

After initialization, I first reassigned each sample to it's nearest centriod. Then, I created new centroids with the mean values of all the samples, which was shown in **Figure 1 :Scatter and Contour of Data**. After that, we repeated these two steps until the centriods kept the same.**Figure 2: Result of my k-means** .The scatter of data was labeled in three group. Initial guess of cluster centres and their different estimates during iterations are marked in blue and the converged result in red.

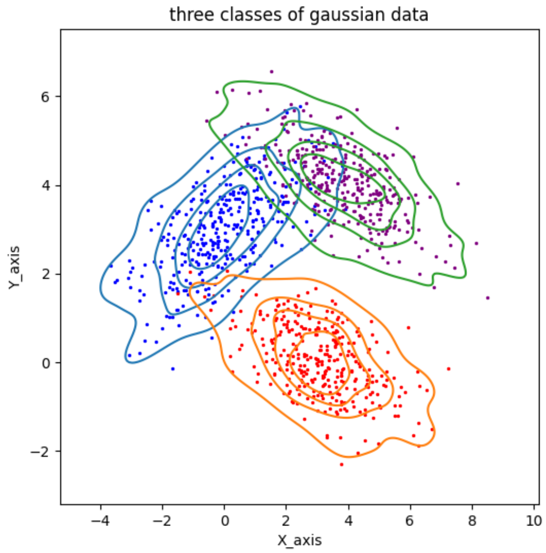


Fig. 1: Scatter and Contour of Data: Data from a mixture Gaussian density

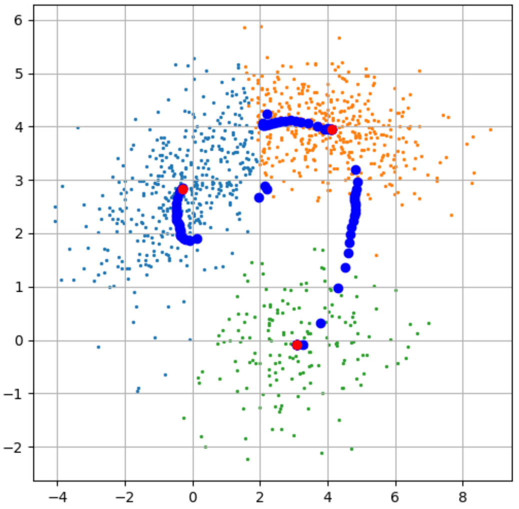


Fig. 2: Result of my k-means:The scatter of data was labeled in three group. Initial guess of cluster centres and their different estimates during iterations are marked in blue and the converged result in red.

B. Contours on the probability density

In this section, I drew contours on the probability density. By comparing the regions associated with each cluser, I found the overlapping area in the raw data **Figure 3:Probability Density of raw data** was larger than that of K-means predicted data **Figure 4:Probability Density of predict data**, which meant the data was grouped more densely after being processed by the K-means algorithm.

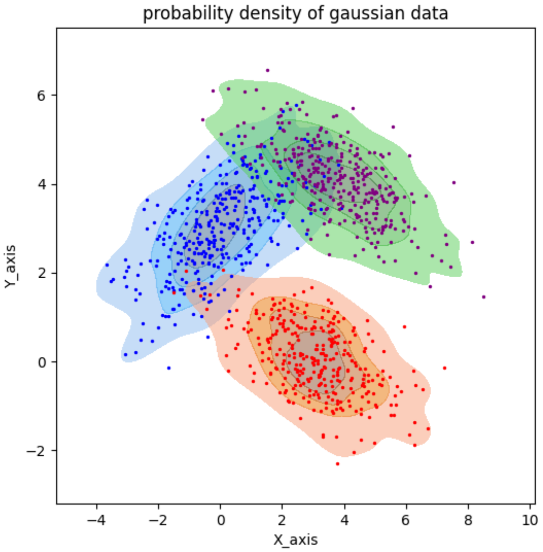


Fig. 3: Probability Density of raw data: Data is labeled in three groups and the five levels of contours indicate the different probabilities in each group.

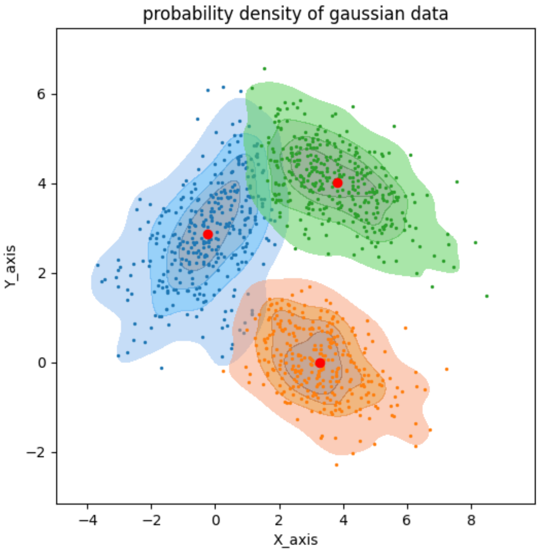


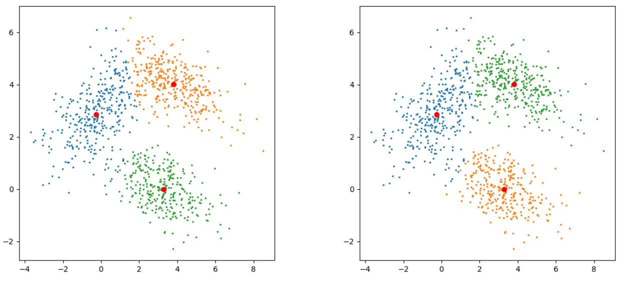
Fig. 4: Probability Density of predicted data: Data is clustered in three groups after implementing K-means algorithm. K is equal to three. The mean values of the data in each group are marked in red.

Besides, I computed the similarity measure of the results of these two clustering models by importing the function adjusted_rand_score from sklearn.metrics and I got **ARI score of 0.7763** as a result.

C. Comparison of my result with the one on sklearn

I compared the results of mine and sklearn from three aspectx:

- Scatters of different groups of data. **Figure5: Scatters of different results**. The scatters were almost the same with the same centroids.



(a) sklearn result

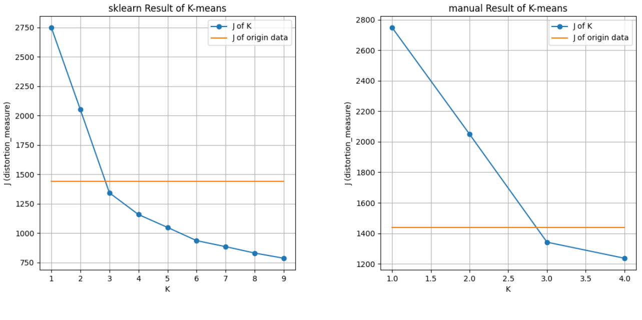
(b) manual result

Fig. 5: Scatters of different results

- J, which is the distortion measure, give by

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_nk\|x_n - \mu_k\|$$

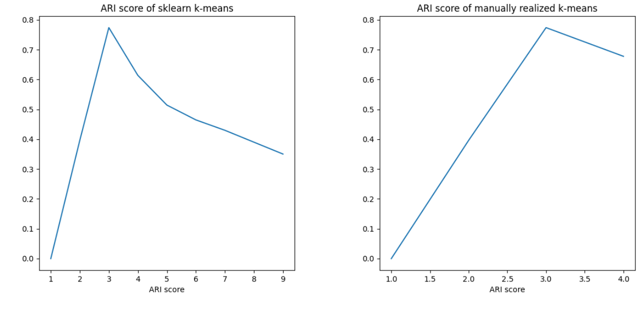
Whether implementing my own method or the sklearn one, I got the same J with the same value of K. **Figure 6: Distortion measure of different values of K** The difference is that sklearn runs faster because there is an optimization of computation in sklearn. In a limited time, I could run more on the tests of K values in sklearn. However, when the result converged, my own method got stuck since the large amount of calculation.
Besides, I found J is a little bit lower than that of raw data at K of three. As the fact that K-means performs best at $K = 3$, so I considered that computing distortion measure could also be used as a method to get a suitable value of K.



(a) sklearn result (b) manual result

Fig. 6: Distortion measure of different values of K: The horizontal axis is the value of K while the vertical axis is the value of J. Distortion measure drops sharply between center of one and center of the appropriate value then it decreases slowly. The target value is marked with a yellow line for better comparison.

- ARI (adjusted Rand index) score ranges from[-1, 1], which ignores permutations and with chance normalization. From **Figure 7:ARI score of different values of K**. In both methods of implementing K-means, I got the same ARI score **0.7763**, which meant the algorithm worked effectively on the data set.



(a) sklearn result (b) manual result

Fig. 7: ARI score of different values of K: The horizontal axis is the value of K while the vertical axis is the value of ARI score.In both figures, ARI score reaches the peak value **0.7763** at K of three and declines as the increase of K.

D. *Examples of the algorithm failing*

Definitely, K-means algorithm is sensitive to the initial guess of the cluster centers and the choice of K. In my implementation, I chose a random cluster center by limiting the mean value to the maximum and minimum of the data.
But there could be some cases I got the same value of mean. In this condition, the data would not converge any more. The K-means algorithm failed when I assigned the same value to the initial cluster centers. And I got zero in ARI score. **Figure 8: The scatter of failure case** , which was the same result when I assigned K with one.

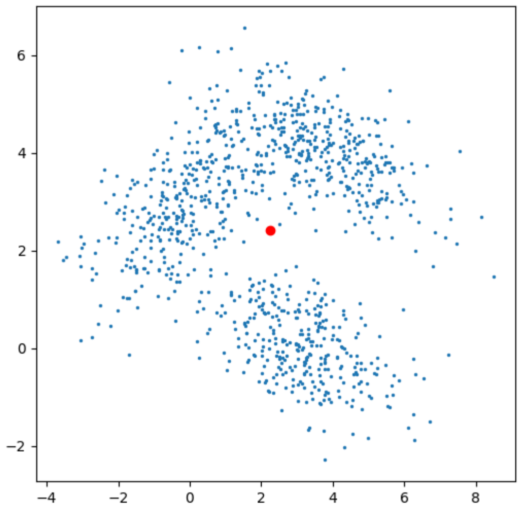
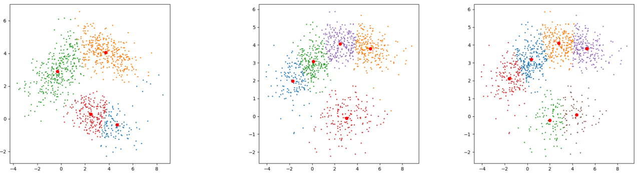


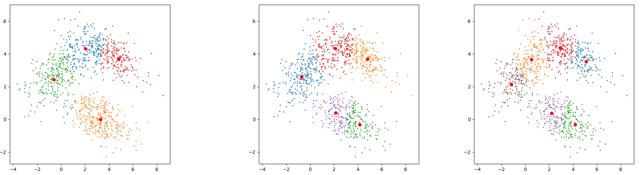
Fig. 8: The scatter of failure case

In addition, by comparing the figures in the first clustering **Figure 9 :The first clustering of K values from 4 to 6** and second clustering **Figure 9 :The second clustering of K values from 4 to 6**, I could see that even with the same number of clusters, the result was sometimes different because of the different initial means.



(a) result of four groups (b) result of five groups (c) result of six groups

Fig. 9: The first clustering of K values from 4 to 6



(a) result of four groups (b) result of five groups (c) result of six groups

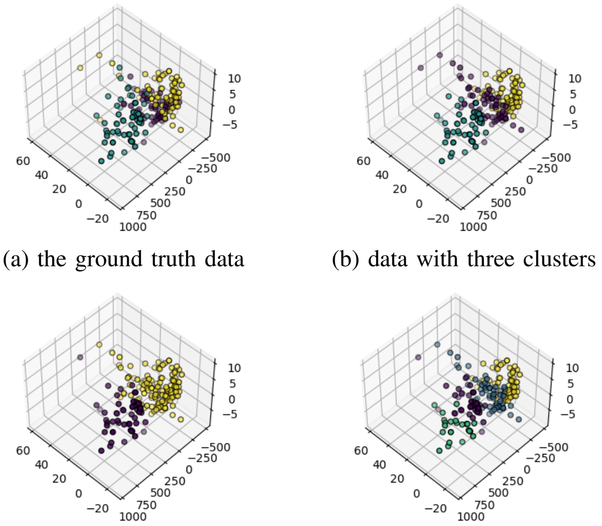
Fig. 10: The second clustering of K values from 4 to 6

ALso, the choice of K matters in K-means algorithm. K-means algorithm does not work at 0. Moreover, when I chose a larger K-value, the distortion measure did drop while the ARI score rose, which meant the choice of K had a significant impact on the performance of the model.

E. Implementing on UCI repository

In this section, I selected **Wine Data set** [1] from UCI repository. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars.The dataset has **178 instances** and **13 attributes**.The analysis determined the quantities of 13 constituents *(Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, Proline found in each of the three types of wines.)* In addition, the data was used with many others for comparing various classifiers. The classes are separable, though only RDA has achieved 100%.

I used PCA (Principle Component Analysis) to project thirteen-dimension data to three dimension data. So I could see the clustering result from the 3D plot.**Figure 11:3D plot view on the result of wine data set clustering**. I found that the data was overlapping in the 3D view of the ground truth data while the predicted groups from K-means were seperable, which meant the predicted data was more dense than the target one.



(a) the ground truth data (b) data with three clusters

(c) data with two clusters (d) data with four clusters

Fig. 11: 3D plot view on the result of wine data set clustering

After that, I implemented k-means model on the data and recorded predicted result at different value of K. To check how well the clusters relate to the targets defined in the dataset, I evaluated the performance of the clustering result in **Adjusted Rand Index**, **Adjusted Mutual Infromation** and **V-measure** respectively.

- ARI score: Adjusted Rand Index is a function that measures the **similarity** of the two assignments.
- AMI score: Adjusted Mutual Information is a function that measures the **agreement** of the two assignments.
- V-measure: V-measure is a intuitive metrics defined by Rosenberg and Hirschberg. [2]. Homogeneity refers each cluster contains only members of a single class while completeness refers all members of a given clas are assigned to the same cluster:

$$V\!-\!measure = \frac{((1+\beta)*homogeneity*completeness}{(\beta*homogeneity+completeness}$$

The score of the k-means model is shown in **Figure 12: Score of different clusters of wine dataset**. I found that the k-means clustering of wine data set does not perform well. When the data was clustered in three groups, ARI score of the model was 0.3711 , AMI score of that was 0.4266 and V-measure score was 0.4287. I supposed that K-mean algorithm performed poorly on a high-dimensional data because of the fact that high-dimensional data was more scattered in space, which was not good for the K-means model to find a centriod.

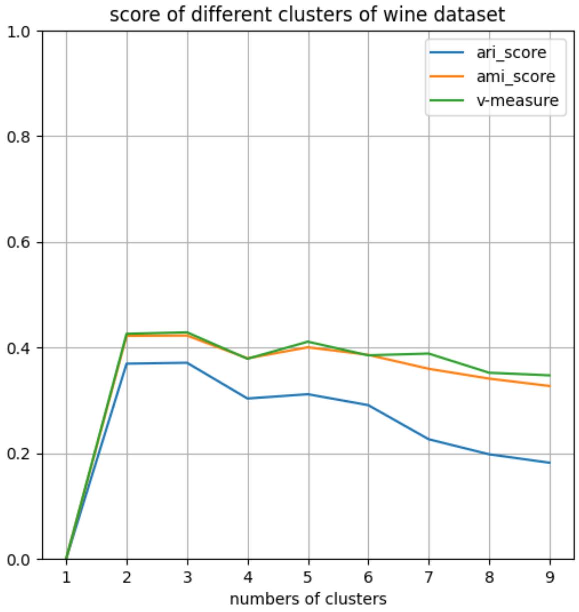


Fig. 12: Score of different clusters of wine data set: The blue line indicated the change of ARI score with K, the yellow one indicated change of AMI score, and the green one indicated v-measure score. All of the lines in the graph reaches a peak when K is equals to two, and the lines are almost flat between the clusters of two and three.

References

[1] PARVUS Forina, M. et al. UCI machine learning repository, 1991.

[2] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.