Machine Learning Report

Andreas Petrou

December 2021

1 Introduction

This report outlines how the Expectation-Maximization algorithm was implemented, in Python, in order to divide a set of flowers into three categories or clusters. It compares the accuracy obtained with K-Means and EM and analyses the number of iterations required for the EM algorithm to converge.

2 Principal Component Analysis

Data provided were in the form of four-dimensional feature vectors. To visualize the data in a graph the dimensions of the feature vectors had to be reduced from 4D into 2D. To achieve this Principal Component Analysis(PCA) for dimensionality reduction was used. Before applying the PCA algorithm to the data the data-set had to be centred. This is because different features have different ranges [1]. For example the range of values sepal length can take differ than the range of values petal width can take. To determine which features to use and which to exclude the explained variance ratio was used. Explained variance is a way to estimate how dissimilar a model is from the actual data. The larger the value of the explained variance the stronger the relationship between the model and the data [2]. The following code calculates the explained variance of each of the four features:

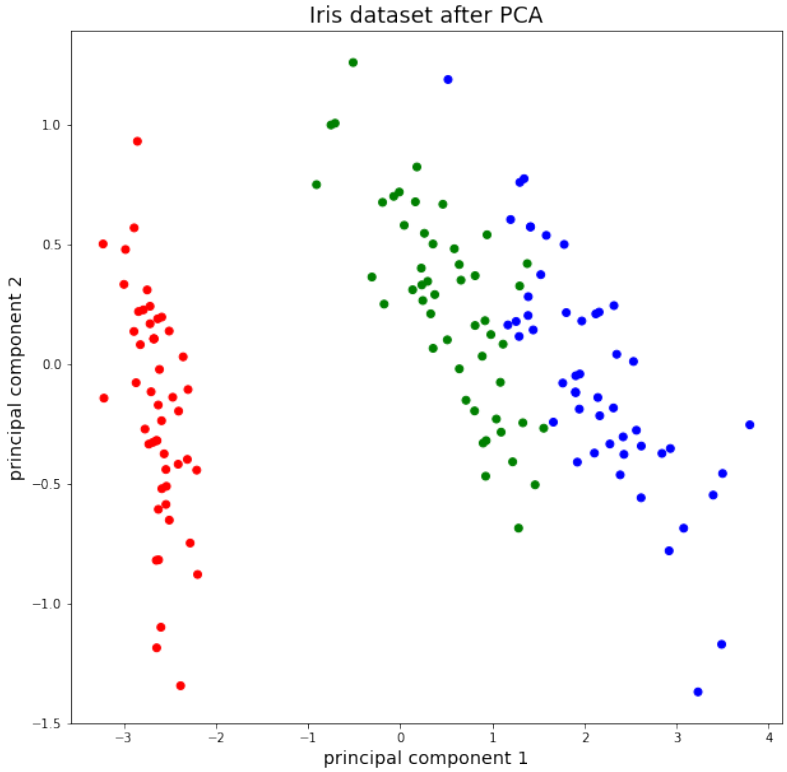
```
U, S, VT = np.linalg.svd(X)

ev = (S ** 2) / 4
ev_ratio = ev / ev.sum()
cumulative_ev = []
for i in range(len(ev_ratio)):
    if i == 0:
        cumulative_ev.append(ev_ratio[i])
    else:
        summation = ev_ratio[i]+cumulative_ev[i-1]
        cumulative_ev.append(summation)
```

After running the code the explained variance for each feature is calculated and the following results were obtained

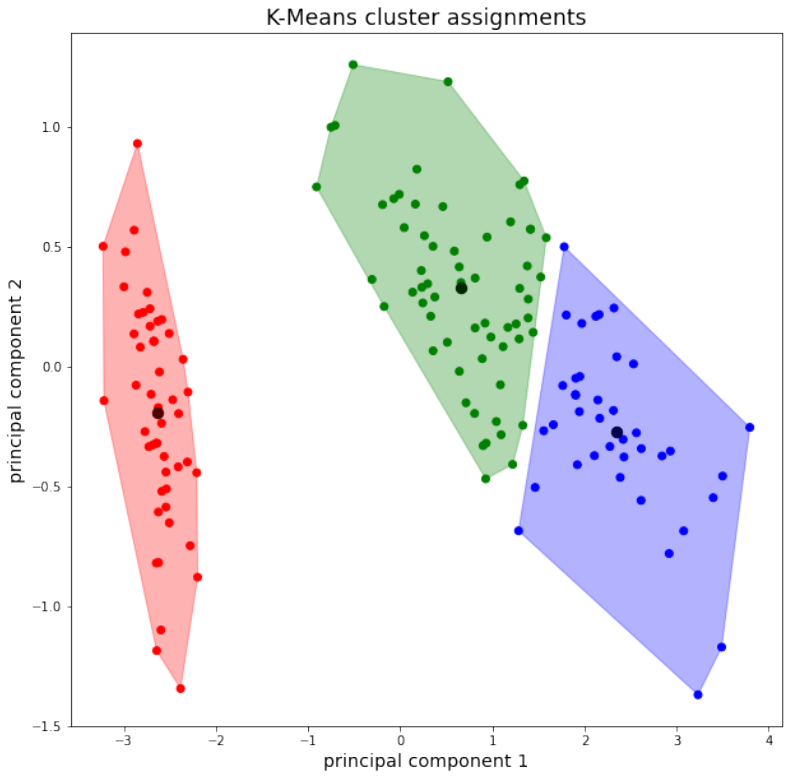
Explained variance: [0.728 0.230 0.0368 0.005]
Cumulative explained variance: [0.728, 0.958, 0.995, 1.000]

The graph obtained after centering the data and projecting it into the two top principal components is portrayed below:



3 K-Means

The first step of the EM algorithm is to initialize a set of parameters known as Θ. This set of parameters include the weight, mean and co-variance of each cluster. To initialise these parameters, the K-means algorithm was used. The cluster shapes after hard assigning data-points to clusters are depicted in the graph below:



4 Expectation-Maximization

After initializing Θ, the EM algorithm consists of two steps. The expectation step and the maximization step. In the expectation step for each data-point x_i we calculate the probability that it belongs to cluster c and then normalize over all clusters [4][5].

$$r_c^i = p(z_i = c|x_i) = \frac{p(z_i = c)p(x_i|z_i = c)}{\sum_{c'} p(z_{c'})p(x_i|z_{c'})} = \frac{\pi_c N(x_i|\mu_c, \Sigma_c)}{\Sigma_{c'}\pi_{c'} N(x_i|\mu_{c'}\Sigma_{c'})} \tag{1}$$

The higher the value of r_c^i the higher the probability that x_i is a member of cluster c. The result of the e-step is to produce a $150x3$ matrix where each row sum up to 1. Then this matrix is used in the Maximization step to update the

parameters of Θ as follows:

$$\pi_c^{t+1} = \frac{1}{n}\sum_{i=1}^{n} r_c^i \tag{2}$$

$$\mu_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i}\sum_{i=1}^{n} r_c^i x_i \tag{3}$$

$$\Sigma_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i}\sum_{i=1}^{n} r_c^i (x_i - \mu_c^{t+1})(x_i - \mu_c^{t+1})^T \tag{4}$$

5 EM convergence

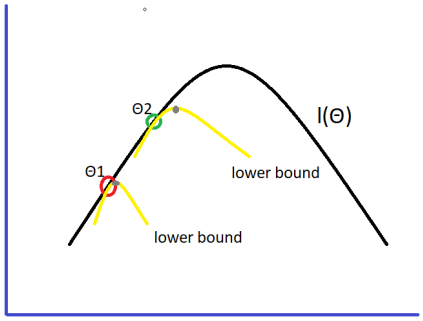
The two steps described in the previous section are repeated in succession until the algorithm converges. It is said that the algorithm converges when the value of the loss function at iteration *t* and *t-1* differ by less than 0.0001. To compute the accuracy of the models the actual labels computed by both K-means and EM had to be converted from numbers ranging from 0 to 2 into the actual type of flower. Then the ground truth labels were compared against predicted labels [5] and the accuracy was calculated as follows:

$$accuracy = \frac{correct\_predictions}{total\_number\_of\_predictions}$$

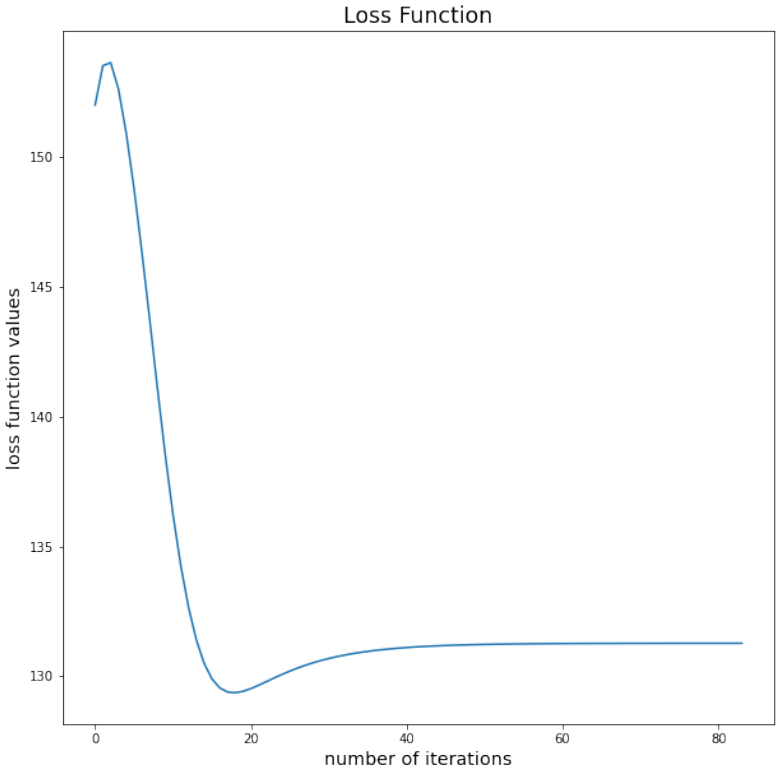
What the algorithm aims to achieve is find a value for theta that maximizes the log-likelihood which can be derived as follows [6]:

$$l(\Theta) = \sum_{i=1}^{n} logp(x_i|\Theta) = \sum_{i=1}^{n} log \sum_{z_i=1}^{k} p(x_i, z_i|\Theta) \tag{5}$$

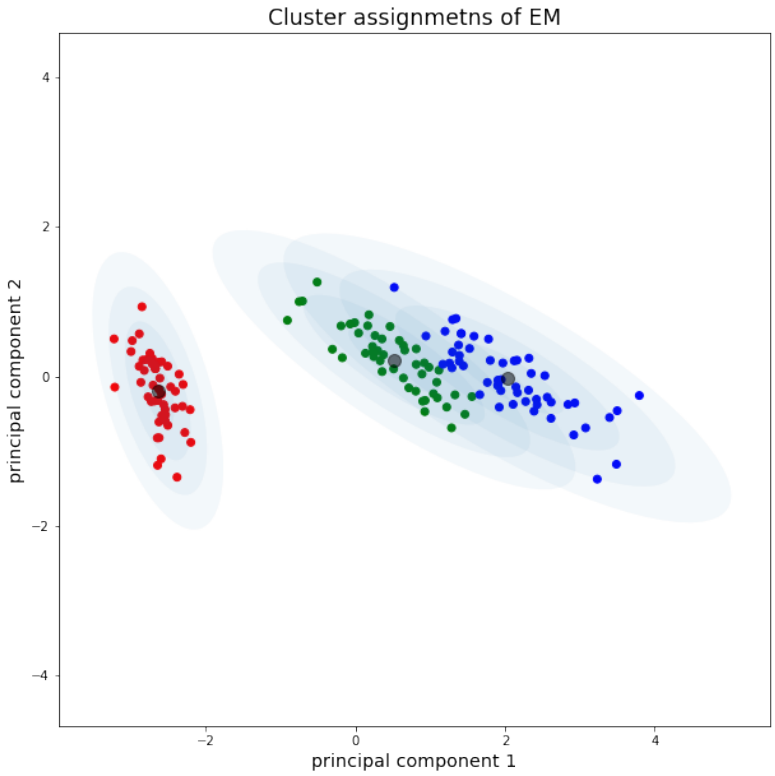
If you have an objective function $l(\Theta)$ and construct a graph for all possible values for Θ then what the EM algorithm attempts to do is at every iteration, after you initialize Θ, the Expectation step will construct a lower bound of log-likelihoods on the line of the graph. Then the Maximization step takes this lower bound and finds its maximum in order to update values of Θ. Initialization of Θ is crucial not only in the accuracy of the model, as the EM algorithm tends to converge in local optima, but also in the number of iterations required to achieve convergence. The graph below depicts a concave function l(Θ):



The expectation step as mentioned above produces a lower bound of log likelihoods and the maximization step updates Theta to the peak of the yellow curves (grey dots). If we run the EM two times. In the first run we initialize Θ to Θ_1 (red dot) and in the second run we initialize Θ to Θ_2 (green dot). Then, in the first run the algorithm will require more steps to converge as it is further away from the peak of the black curve.When the assignment of Θ produces a value of l(Θ) equal to the peak of the black curve then the peak of the lower bound produced by the expectation step is equal to the peak of the black curve and the assignment of Θ does not change hence the algorithm converges. What is worth being noted is that the lower bound has two properties. First, it is below the curve and second, it intersects with the black curve at the current assignment of Θ. In the above graph the first lower bound intersect with the main curve at Θ_1 and the second lower bound intersects at Θ_2 [6]. For our program, EM required 84 steps to converge and a local minimum was found during the 18th iteration.



In addition, the shapes of the clusters after the 18th iteration (iteration where local minimum was achieved) looked like this:



6 Model accuracy and conclusion

Like EM, K-means can also get stuck in local optima. It uses the Euclidean distance between data-points and cluster centroids (means) to determine in which cluster that particular data-point belongs to. Similarly to EM, the number of iterations of K-Means, heavily depends on the initialization of cluster centroids. Both algorithms can get unstuck from local optima and attempt to find the global optimum by initializing their parameters several times. Running the K-means on the Iris data-set gave an accuracy of 88.67% and sum of squared errors 63.87 while EM gave an accuracy of 98%. This does not necessarily mean that EM is a better clustering algorithm than K-means as the given data-set is structured in such a way that it favours EM. This is because some data-points are closer to centroids of clusters which they do not belong to. As a matter of fact, it has been shown that in some scenarios K-means is capable of outperforming EM in terms of accuracy [7].

References

[1] Lakshmanan S. *How, When, and Why Should You Normalize / Standardize / Rescale Your Data?*
https://towardsai.net/p/data-science/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff#:~:text=For%20machine%20learning%2C%20every%20dataset,when%20features%20have%20different%20ranges.&text=So%20we%20normalize%20the%20data,variables%20to%20the%20same%20range.
[Accessed 03/12/2021]

[2] Statistics How To. *Explained Variance / Variation*
https://www.statisticshowto.com/explained-variance-variation/
[Accessed 03/12/2021]

[3] Sanchez V. *Clustering: K-means & soft K-means.* [Lecture] University of Warwick. October 2021.

[4] Ihler A. *Clustering (4): Gaussian Mixture Models and EM*
https://www.youtube.com/watch?v=qMTuMa86NzU&t=389s
[Accessed 21/11/2021]

[5] Bhattacharyya S. *Latent Variables & Expectation Maximization Algorithm*
https://towardsdatascience.com/latent-variables-expectation-maximization-algorithm-fb15c4e0f32c
[Accessed 03/12/2021]

[6] Ng A. *Lecture 14 - Expectation-Maximization Algorithms — Stanford CS229: Machine Learning (Autumn 2018)*
https://www.youtube.com/watch?v=rVfZHWTwXSA&t=3359s
[Accessed 04/12/2021]

[7] Heo J, Kang S M, Jung G Y. *Clustering performance comparison using K-means and expectation maximization algorithms*
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4433949/
[Accessed 04/12/2021]