FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Machnine Learning (PDEEC0049 : 15-782PP)

Homework 5

António Damião das Neves Rodrigues (200400437 : 700098386)

December 24, 2013

1 PROBLEM 1

1.1

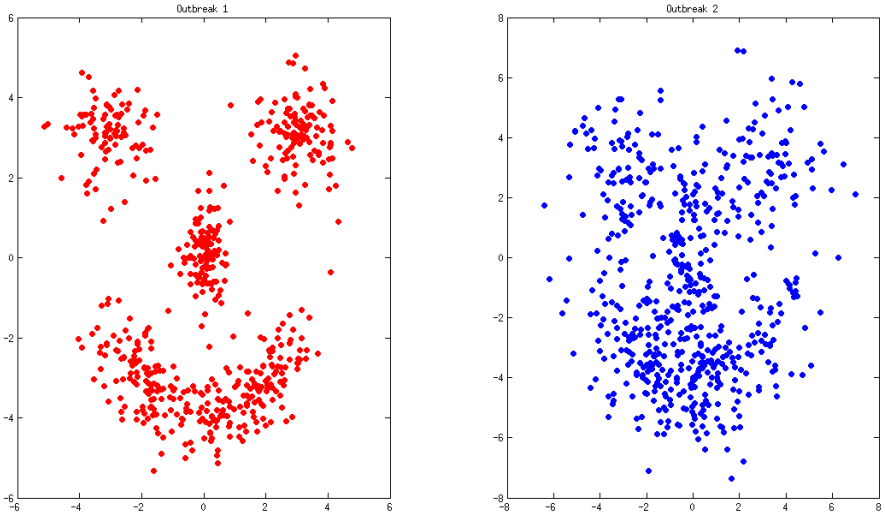
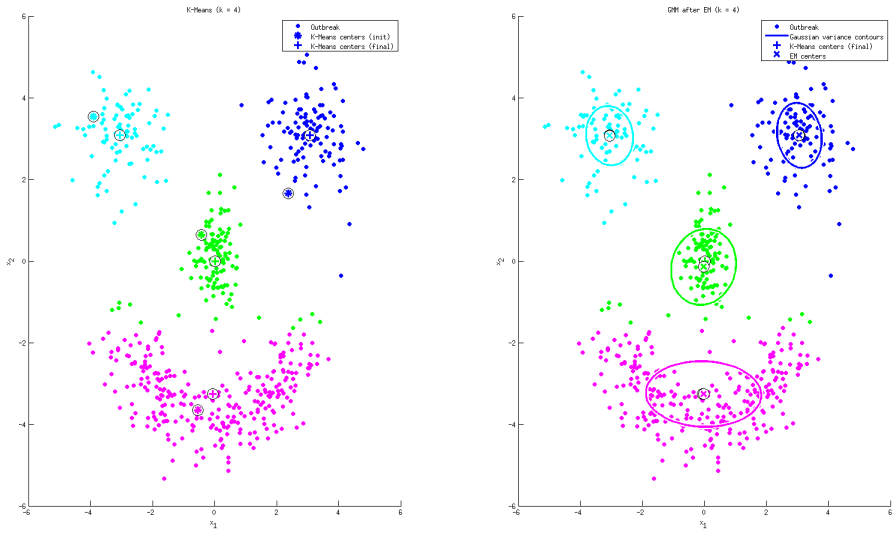


Figure 1.1: Geographical spread of infected computers, for the two virus outbreaks.

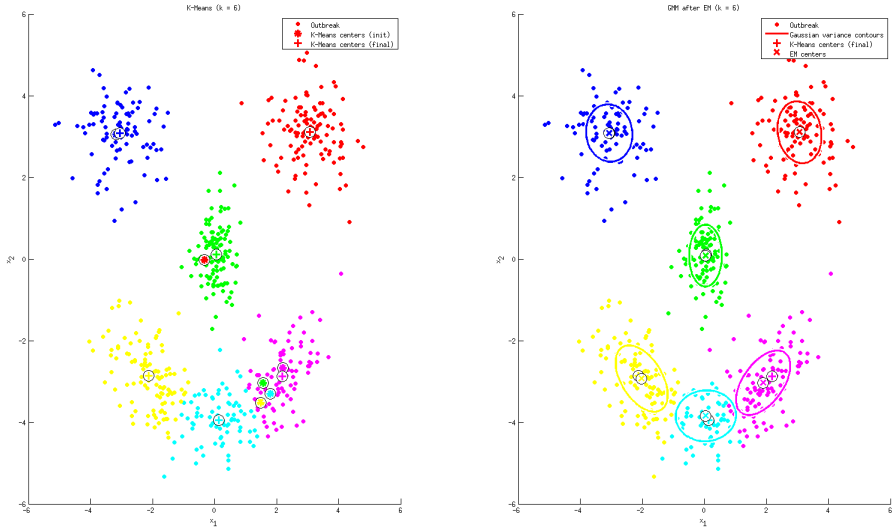
1.2

Use the MATLAB code given as attachment (`problem1` folder) files `kmeans.m` and `em.m` for implementation details. Follow the comments on the code for details and reasoning. The MATLAB file `problem1.m` was used for testing the implementations (in that file you may choose the values of the dataset and K to test).

Figure 1.2 shows the partial result of running `problem1_2.m`, for $K = 4$ and $K = 6$, for the `joker1` dataset. Notice that the final cluster centers of K-Means are used as the initial means for the Gaussiam Mixture Model (GMM), before running the Expected Maximization (EM) algorithm for GMMs.



(a)



(b)

Figure 1.2: Clustering after running K-Means and the EM algorithm for GMMs, for the `joker1` dataset, $K = 4$ (a) and $K = 6$ (b).

1.3

Use the MATLAB code given as attachment (`problem1` folder) files `problem1_3.m` and `cluster.m` for implementation details. Follow the comments on the code for details and reasoning. The MATLAB file `problem1_3.m` was used for testing the implementations.

The script `problem1_3.m` generates a series of subplot groups, for each {K,`joker*`} combination. See Figure 1.3 for an example.

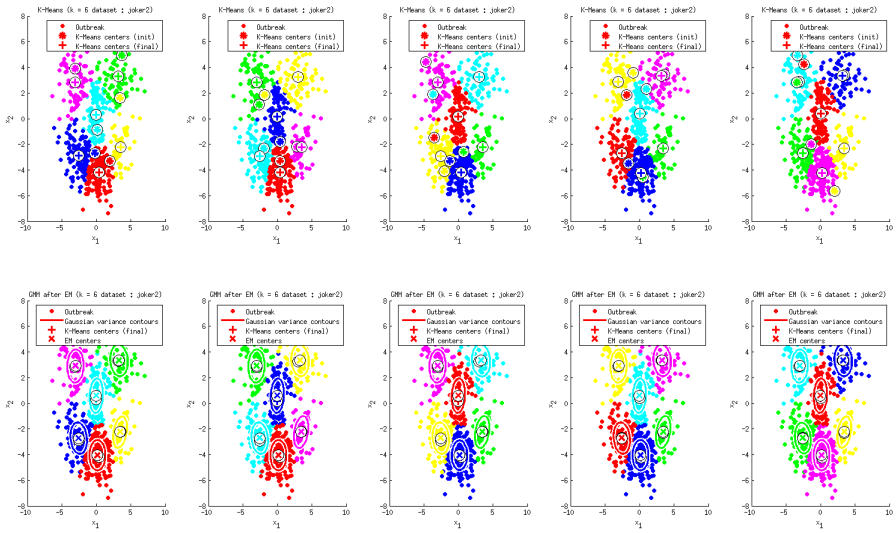


Figure 1.3: Example of output of the `problem1_3.m` MATLAB script, for $K = 6$, dataset `joker2`. Each column corresponds to a different group of initial centers, the final cluster centers of K-Means are used as the initial means for the GMM approach.

2 Problem 2

Use the MATLAB code given as attachment (`problem2` folder) files `hmmTest.m` for implementation details. Follow the comments on the code for details and reasoning.

Table 2.1 shows the computed probabilities for the given sequence, using both forward and backward algorithms (both compute the same value).

Algorithm	$P(x\|M)$
Forward	0.094205886087188
Backward	0.094205886087188

Table 2.1: Computed probabilities for the sequence of length 10 in Problem 2, given the provided Hidden Markov Model M, using both forward and backward algorithms.

3 Problem 3

3.1

Assumptions:

1. The WSN is formed by a $N \times N$ square matrix of nodes W. Therefore, there are N^2 nodes in the WSN and each position (i, j) of W is occupied by a wireless node.
2. When a node in the WSN transmits a message, all other nodes receive the message, i.e. messages are broadcast.
3. All nodes have memory capabilities to store the state of the matrix W, in which each position (i, j) of W holds the value(s) for each variable v sensed by the WSN node at position (i, j).
4. A WSN node can transmit any one or more types of data it knows in one message or a function of this data.
5. Every node is aware of its (i, j) coordinates within W.
6. All nodes are aware of their inclusion in a $N \times N$ square matrix W, with each position accessed via coordinates (i, j).
7. Among the N^2 nodes of the WSN, there is a master node M, with the capability of starting network procedures.
8. In the following descriptions, the indexing of arrays starts at '0', e.g. the first element of an array F would be $F[0]$.

Algorithm:

1: Compute factors of N. Store them in ascending order in $F = \{1, ..., N\}$.
2: $k \leftarrow 1$
3: $T \leftarrow M$
4: **for** $\frac{N}{F[k]} \geq S$ **do**
5: Divide W in $F[k]^2$ squares of side $\frac{N}{F[k]}$.
6: **for all** unvisited squares of side $\frac{N}{F[k]}$ **do**

7: Node T sends message with format $\{(i,j);\text{value};(i',j'\}$. The value (i',j') is a random position within one of the unvisited $F[k]^2$ squares, excluding (i,j) positions already known in W.
8: All nodes update the matrix W with the contents of the message from node T.
9: $T \leftarrow \text{node}(i',j')$
10: **end for**
11: Determine the min. and max. values of the known nodes in W, W_{min} and W_{max}.
12: Set all unknown (i,j) slots of W to W_{min}.
13: Save new positions (i,j) of the local maxima (peaks) of W in the **means** array U.
14: $k \leftarrow k+1$
15: $T \leftarrow$ any unvisited node in W
16: **end for**
17: **for all** positions (i,j) in U **do**
18: From the K^2-1 neighbor positions of U_n, those which are unknown in matrix W send messages in the format $\{(i,j);\text{value}\}$.
19: All nodes update the matrix W with the contents of the message from the neighbor nodes.
20: **end for**
21: Save new positions (i,j) of the local maxima (peaks) of W in the **means** array U.

Explanation:

The first part of the algorithm (lines 4 to 16) consists in sampling the sensor values in a distributed manner. The precision of the sampling — and the number of exchanged messages — can be tuned via the S parameter (the minimum side size of the square partitions to apply in $N \times N$ matrix of sensors). With $S = 1$, we would get the complete knowledge of the dataset, i.e. N^2 exchanged messages.

The second part (lines 17 to 21) attempts to gather more knowledge around the pre-selected means U of the matrix W, by retrieving the values of its neighbors. The neighborhood structure is a square of size K^2 (also tunable), centered around each element of U.

After this sampling phase, the idea would be to use EM for GMMs in the regular way, using the elements of U as the initial means, to determine the remaining parameters of the GMM.

The cost in messages of the algorithm would be: $(\frac{N}{S})^2 + ((\text{size}(U) \times (K^2-1)) - A)$. Note that the second additive term varies with A, the number of neighbors of the nodes in U already known at the time of running the second phase of the algorithm. E.g. for $S = 1$, we would know the entire dataset after the first phase of the algorithm, so the values of all K^2-1 neighbors within the `for` cycle in line 17 would be known, therefore $(\text{size}(U) \times (K^2-1) - A) = 0$, so, in the worst case, this method would also cost N^2 messages.

3.2

Regarding the algorithm presented in Section 3.1, the knowledge of a global maximum would allow it to start similarly to its second part: (1) determine one of initial means U_n (a peak) and (2) the K^2-1 neighborhood of U_n.

As the proposed algorithm already tries to gather $K^2 - 1$ values around or near that peak, that knowledge would not contribute to reduce the number of exchanged messages.