K-means, GMM and HMM

Thomas Schatz

10 février 2017

1 Introduction

The text files *EMGaussienne.data* and *EMGaussienne.test* contain data $(z_1, z_2, ...z_n)$ where each z_i is in $\mathbb{R}^2$. The objective of this TD is to fit a series of increasingly more sophisticated models to this data. We will test K-means clustering, a mixture of diagonal-covariance Gaussians, a mixture of full-covariance Gaussians and Hidden Markov Models with Gaussian emission probabilities. All the models will be fitted using optimization heuristics based on alternating between assigning each data point to a cluster given current estimates for the model parameters (E step) and computing the optimal values for the model parameters given current assignments of the data points (M step). For K-means we use Lloyd's algorithm `https://en.wikipedia.org/wiki/Lloyd's_algorithm`, for Gaussian Mixtures we use the Expectation Maximization algorithm `https://en.wikipedia.org/wiki/Expectation-maximization_algorithm`, for Hidden Markov Models we use the Forward Backward algorithm `https://en.wikipedia.org/wiki/Forward-backward_algorithm`. For each model we will choose a number K of components equal to 4, unless it is explicitly specified otherwise.

2 Preliminary

1. Load and graphically visualize the training and test data.

3 K-means

1. Fit a K-means clustering to the training data using Lloyd's algorithm.
2. Compute cluster assignations for the training and test data.
3. Graphically represent the training and test data with different colors for each cluster. Plot also on the same graph the centers of the obtained clusters.
4. Train a model for different initializations and compute each time the distortion measure on the training and test data.
5. **Bonus question 1.** Read : `http://normaldeviate.wordpress.com/2012/09/30/the-remarkable-k-means` and implement the initialization scheme described there. Recompute the distortion measure for various runs and compare with the results from the previous question.
6. **Bonus question 2.** Read : `http://papers.nips.cc/paper/6478-fast-and-provably-good-seedings-for-k-means` and implement the initialization scheme described there. Recompute the distortion measure for various runs and compare with the results from the previous question.
7. **Bonus question 3.** How could you have chosen K if you had no a priori ? Try your proposed solution(s).

4 Mixture of Gaussians with covariance proportional to the identity matrix

1. Implement the Expectation Maximization algorithm for Gaussian mixtures with covariance matrices proportional to the identity matrix. Initialize the parameters with the results from the previous section.
2. Compute cluster assignations for the training and test data.
3. Graphically represent the training and test data with different colors for each cluster. Plot also on the same graph the means and variance of the mixture components. To represent the variance, one possibility is to draw an iso-probability ellipse containing a fixed percentage of the probability mass (e.g. 95%).
4. Compare the results obtained with different initializations and comment.
5. Compute the log-likelihood of the model on the training and test data.

5 Mixture of Gaussians

1. Answer the same questions as in the previous section, but for full-covariance Gaussian mixtures. Compare the results with those from the previous section.

6 Hidden Markov Model

Consider now the data as a time series $(z_1, z_2, ..., z_T)$. We will model it as a HMM with 4 hidden states and transition matrix $a \in \mathbb{R}^{4\times4}$. We will note $(q_t)_{1\leq t\leq T}$ the sequence of the hidden states and $(u_t)_{1\leq t\leq T}$ the sequence of the observable states. We will consider Gaussian emission probabilities : $u_t \mid q_t \sim \mathcal{N}(\mu_{q_t}, \Sigma_{q_t})$.

1. Implement the α and β recursions from the Forward Backward algorithm to estimate $p(q_t|u_1, u_2, ..., u_T)$ and $p(q_t, q_{t+1}|u_1, u_2, ..., u_T)$.
2. Using the code from question 1, implement the Baum-Welch algorithm for learning the parameters of the HMM. Initialize the parameters with the results from the previous section.
3. Using the code from question 1, implement the Viterbi algorithm for classifying new data with the learnt HMM.
4. Classify and represent the train and test data.
5. Compute the log-likelihood of the train and test data. Compare the results with those from Section 5.

7 Conclusion

1. Compare and comment the results obtained with the different methods.
2. Can you suggest other methods that we could use to model this dataset ?