Homework 1

Ayman YACHAOUI
Graphical Models
Master Data & Knowledge
Telecom ParisTech
ayman.yachaoui@telecom-paristech.fr

1 Learning in discrete graphical models

Let's write the observations for x and z respectively $(x_i)_{1\leq i\leq N} \in \mathrm{J}1, K\mathrm{K}^N$ and $(z_i)_{1\leq i\leq N} \in \mathrm{J}1, M\mathrm{K}^N$. We have the following information:

$$p_\pi(z = m) = \pi_m$$
$$p_\theta(x = k \mid z = m) = \theta_{mk}$$
$$\sum_{m=1}^{M} \pi_m = 1$$
$$\sum_{k=1}^{K} \theta_{mk} = 1$$
$$\sum_{m=1}^{M}\sum_{k=1}^{K} \pi_m\theta_{mk} = 1$$

Then we compute the opposite log-likelihood:

$$\begin{aligned}
-\ell(\pi,\theta) &= -\log\left(\prod_{i=1}^{N} p_{\pi,\theta}(z = z_i, x = x_i)\right) \\
&= -\sum_{i=1}^{N} \log\left(p_\pi(z = z_i)p_\theta(x = x_i \mid z = z_i)\right) \\
&= -\sum_{i=1}^{N} \log \pi_{z_i} - \sum_{i=1}^{N} \log \theta_{z_i x_i}
\end{aligned}$$

Using the notation:

$$\forall m \in \mathrm{J}1, M\mathrm{K}, n_m = \sum_{i=1}^{N} 1_{z_i=m}$$
$$\forall (m,k) \in \mathrm{J}1, M\mathrm{K} \times \mathrm{J}1, K\mathrm{K}, n_{mk} = \sum_{i=1}^{N} 1_{z_i=m, x_i=k}$$

We simplify the opposite log-likelihood:

$$-\ell(\pi,\theta) = -\sum_{m=1}^{M} n_m \log \pi_m - \sum_{m=1}^{M}\sum_{k=1}^{K} n_{mk} \log \theta_{mk}$$

This is a convex function that we need to minimize, thus we compute the Lagrangian:

$$\mathcal{L}(\pi, \theta, \lambda, \mu, (\gamma_m)_{1 \leq m \leq M}) = -\sum_{m=1}^{M} n_m \log \pi_m - \sum_{m=1}^{M} \sum_{k=1}^{K} n_{mk} \log \theta_{mk} + \lambda \left(1 - \sum_{m=1}^{M} \pi_m\right) + \mu \left(1 - \sum_{m=1}^{M} \sum_{k=1}^{K} \pi_m \theta_{mk}\right) + \sum_{m=1}^{M} \gamma_m \left(1 - \sum_{k=1}^{K} \theta_{mk}\right)$$

Then by using the fact that the partial derivative of the Lagrangian must be nul for the minimum we have:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \pi_m}(\hat{\pi}_m, ...) = 0 &= -\frac{n_m}{\hat{\pi}_m} - \lambda - \mu \sum_{k=1}^{K} \hat{\theta}_{mk} \\ &= -\frac{n_m}{\hat{\pi}_m} - \lambda - \mu \\ \frac{\partial \mathcal{L}}{\partial \theta_{mk}}(..., \hat{\theta}_{mk}, ...) = 0 &= -\frac{n_{mk}}{\hat{\theta}_{mk}} - \mu \hat{\pi}_m - \gamma_m \end{aligned}$$

Therefore by using the probability distribution condition on $\hat{\theta}$:

$$\begin{aligned} 1 &= \sum_{k=1}^{K} \hat{\theta}_{mk} \\ &= -\sum_{k=1}^{K} \frac{n_{mk}}{\mu \hat{\pi}_m + \gamma_m} \\ &= -\frac{n_m}{\mu \hat{\pi}_m + \gamma_m} \end{aligned}$$

And by reinjecting:

$$\hat{\theta}_{mk} = -\frac{n_{mk}}{\mu \hat{\pi}_m + \gamma_m} = \frac{n_{mk}}{n_m}$$

We follow the same process with $\hat{\pi}$:

$$\begin{aligned} 1 &= \sum_{m=1}^{m} \hat{\pi}_m \\ &= -\sum_{m=1}^{m} \frac{n_m}{\lambda + \mu} \\ &= -\frac{N}{\lambda + \mu} \end{aligned}$$

Therefore:

$$\hat{\pi}_m = -\frac{n_m}{\lambda + \mu} = \frac{n_m}{N}$$

To summarize we have the maximum likelihood estimators for π and θ:

$$\begin{aligned} \hat{\pi}_m &= \frac{n_m}{N} \\ \hat{\theta}_{mk} &= \frac{n_{mk}}{n_m} \end{aligned}$$

2 Linear classification

1 Generative model (LDA)

(a) Let's write the observations for x and y respectively $(x_i)_{1\leq i\leq N} \in \mathbb{R}^{2\times N}$ and $(z_i)_{1\leq i\leq N} \in \{0,1\}^N$. We have the following information:

$$p_\pi(y=k) = \pi^k (1-\pi)^{1-k}$$
$$p_{\mu_i,\Sigma}(x=z \mid y=i) = \frac{1}{\sqrt{(2\pi)^2 |\Sigma|}} \exp\left(-\frac{1}{2}(z-\mu_i)^\intercal \Sigma^{-1}(z-\mu_i)\right)$$

Then we compute the opposite log-likelihood:

$$\begin{aligned}
-\ell(\pi,\mu_0,\mu_1,\Sigma) &= -\log\left(\prod_{i=1}^N p_{\pi,\mu_0,\mu_1,\Sigma}(x=x_i, y=y_i)\right) \\
&= -\sum_{i=1}^N \log\left(p_\pi(y=y_i) p_{\mu_0,\mu_1,\Sigma}(x=x_i \mid y=y_i)\right) \\
&= -\sum_{i=1}^N y_i \log\pi - \sum_{i=1}^N (1-y_i)\log(1-\pi) + N\log 2\pi \\
&+ \frac{N}{2}\log|\Sigma| + \sum_{i=1}^N \left(\frac{1}{2}(x_i-\mu_{y_i})^\intercal \Sigma^{-1}(x_i-\mu_{y_i})\right)
\end{aligned}$$

Using the notation:

$$n = \sum_{i=1}^N y_i$$

We simplify the opposite log-likelihood:

$$\begin{aligned}
-\ell(\pi,\mu_0,\mu_1,\Sigma) = -n\log\pi - (N-n)\log(1-\pi) + N\log 2\pi + \frac{N}{2}\log|\Sigma| \\
+ \frac{1}{2}\sum_{\substack{i=1\\ y_i=0}}^N (x_i-\mu_0)^\intercal \Sigma^{-1}(x_i-\mu_0) + \frac{1}{2}\sum_{\substack{i=1\\ y_i=1}}^N (x_i-\mu_1)^\intercal \Sigma^{-1}(x_i-\mu_1)
\end{aligned}$$

We need to minimize this function. We observe that there are two independents function, one of π and one of (μ_0,μ_1,Σ).
First we minimize $f : \pi \mapsto -n\log\pi - (N-n)\log(1-\pi)$ and we find:

$$\hat{\pi} = \frac{n}{N}$$

Then we minimize g defined bellow with respect to μ_0 and μ_1.

$$g : (\mu_0,\mu_1) \mapsto \sum_{\substack{i=1\\ y_i=0}}^N (x_i-\mu_0)^\intercal \Sigma^{-1}(x_i-\mu_0) + \sum_{\substack{i=1\\ y_i=1}}^N (x_i-\mu_1)^\intercal \Sigma^{-1}(x_i-\mu_1)$$

As it consists of two quadratic and thus convex function, we find the point where the derivatives are nul.

$$\begin{aligned}
\frac{\partial g}{\partial \mu_0}(\hat{\mu}_0, \mu_1) = 0 &= -2 \sum_{\substack{i=1 \\ y_i=0}}^{N} \Sigma^{-1}(x_i - \hat{\mu}_0) \\
&= -2\Sigma^{-1} \left(\left(\sum_{\substack{i=1 \\ y_i=0}}^{N} x_i \right) - (N-n)\,\hat{\mu}_0 \right) \\
\frac{\partial g}{\partial \mu_1}(\mu_0, \hat{\mu}_1) = 0 &= -2 \sum_{\substack{i=1 \\ y_i=1}}^{N} \Sigma^{-1}(x_i - \hat{\mu}_1) \\
&= -2\Sigma^{-1} \left(\left(\sum_{\substack{i=1 \\ y_i=1}}^{N} x_i \right) - n\hat{\mu}_1 \right)
\end{aligned}$$

Thus we find the two maximum likelihood estimators for μ_0 and μ_1:

$$\begin{aligned}
\hat{\mu}_0 &= \frac{1}{N-n} \sum_{\substack{i=1 \\ y_i=0}}^{N} x_i \\
\hat{\mu}_1 &= \frac{1}{n} \sum_{\substack{i=1 \\ y_i=1}}^{N} x_i
\end{aligned}$$

Now we consider the minimisation of the function:

$$h : \Sigma^{-1} \mapsto -N \log |\Sigma^{-1}| + \sum_{i=1}^{N} (x_i - \hat{\mu}_{y_i})^\intercal \, \Sigma^{-1} \, (x_i - \hat{\mu}_{y_i})$$

First we observe that:

$$\begin{aligned}
h(\Sigma^{-1}) &= -N \log |\Sigma^{-1}| + \operatorname{tr} \left(\sum_{i=1}^{N} (x_i - \hat{\mu}_{y_i})^\intercal \, \Sigma^{-1} \, (x_i - \hat{\mu}_{y_i}) \right) \\
&= -N \log |\Sigma^{-1}| + \operatorname{tr} \left(\Sigma^{-1} \sum_{i=1}^{N} (x_i - \hat{\mu}_{y_i}) \, (x_i - \hat{\mu}_{y_i})^\intercal \right)
\end{aligned}$$

And using:

$$\begin{aligned}
\frac{\partial}{\partial U} \log |U| &= U^{-\intercal} \\
\frac{\partial}{\partial U} \operatorname{tr}(UV) &= V^\intercal
\end{aligned}$$

Taking the derivative with respect to Σ^{-1} we have:

$$\frac{\partial h}{\partial \Sigma^{-1}}(\hat{\Sigma}^{-1}) = 0 = -N\hat{\Sigma}^\intercal + \sum_{i=1}^{N} (x_i - \hat{\mu}_{y_i}) \, (x_i - \hat{\mu}_{y_i})^\intercal$$

Thus to summarize:

$$\hat{\pi} = \frac{n}{N}$$

$$\hat{\mu}_0 = \frac{1}{N-n} \sum_{\substack{i=1 \\ y_i=0}}^{N} x_i$$

$$\hat{\mu}_1 = \frac{1}{n} \sum_{\substack{i=1 \\ y_i=1}}^{N} x_i$$

$$\hat{\Sigma} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{\mu}_{y_i}) (x_i - \hat{\mu}_{y_i})^\intercal$$

(b) Finally we have the conditional distribution:

$$\begin{aligned}
p(y=1 \mid x) &= \frac{p(y=1)p(x \mid y=1)}{p(x)} \\
&= \frac{p(y=1)p(x \mid y=1)}{p(x \mid y=1) + p(x \mid y=0)} \\
&= \frac{\hat{\pi}\mathcal{N}_{\hat{\mu}_1,\hat{\Sigma}}(x)}{\mathcal{N}_{\hat{\mu}_1,\hat{\Sigma}}(x) + \mathcal{N}_{\hat{\mu}_0,\hat{\Sigma}}(x)} \\
&= \frac{\hat{\pi}}{1 + \frac{\mathcal{N}_{\hat{\mu}_0,\hat{\Sigma}}(x)}{\mathcal{N}_{\hat{\mu}_1,\hat{\Sigma}}(x)}} \\
&= \frac{\hat{\pi}}{1 + \exp\left(-\frac{1}{2}(x-\hat{\mu}_0)^\intercal \hat{\Sigma}^{-1}(x-\hat{\mu}_0) + \frac{1}{2}(x-\hat{\mu}_1)^\intercal \hat{\Sigma}^{-1}(x-\hat{\mu}_1)\right)} \\
&= \frac{\hat{\pi}}{1 + \exp\left(-(\hat{\mu}_1-\hat{\mu}_0)^\intercal \hat{\Sigma}^{-1}x - \frac{1}{2}\hat{\mu}_0^\intercal\hat{\Sigma}^{-1}\hat{\mu}_0 + \frac{1}{2}\hat{\mu}_1^\intercal\hat{\Sigma}^{-1}\hat{\mu}_1\right)}
\end{aligned}$$

Which, corresponds to the logistic function multiplied by a factor $\hat{\pi}$.

(c) Finally we plot the three different distributions for the datasets A, B and C plus the normal distribution means μ_0 and μ_1 and the equiprobability line defined by $p(y=1 \mid x) = p(y=0 \mid x) = \frac{1}{2}$. The code for obtaining those figures can be found in the file ***generativemodel.py***.

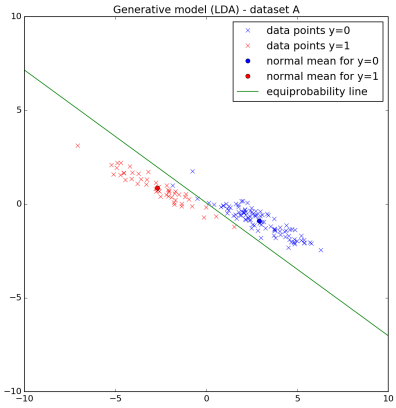

Figure 1: Generative model set A.

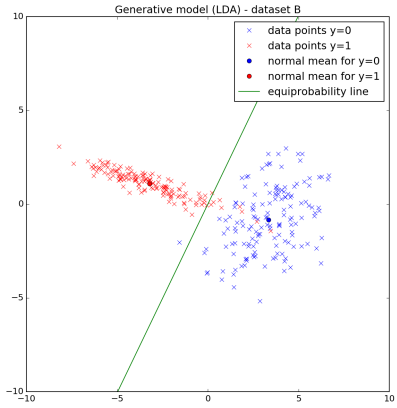

Figure 2: Generative model set B.

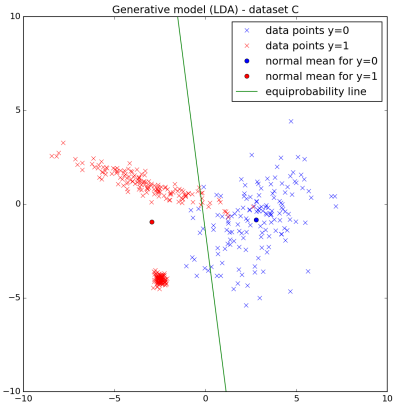

Figure 3: Generative model set C.

2 Logistic regression

(a) The code for obtaining the numerical values of the parameters learnt can be found in the file ***logisticregression.py***. It is pretty straightforward apart from the fact that we first try the regular IRLS algorithm, and if

it does not converge (for instance because the data is not linearly separable), we try a normalized version of the IRLS algorithm. Note that as the data $x^{(k)}_{1\leq k\leq n}$ was normalized into $\tilde{x}^{(k)}_{1\leq k\leq n}$ before running the algorithm thus we include the two parameters $\bar{x}$ and $\sigma \succeq 0$ defined by:

$$\bar{x}_i = \sum_{k=1}^{n} x_i^{(k)}$$

$$\sigma_i^2 = \frac{1}{n}\sum_{k=1}^{n}\left(x_i^{(k)} - \bar{x}_i\right)^2$$

$$\tilde{x}^{(k)} = \begin{pmatrix} \frac{x_0^{(k)} - \bar{x}_0}{\sigma_0} \\ \frac{x_1^{(k)} - \bar{x}_1}{\sigma_1} \\ 1 \end{pmatrix}$$

Finally this gives us the numerical values of the parameter $\hat{w}$ for a precision of 10^{-6} that can be found below.

Dataset	A	B	C
Algorithm	normalized	regular	regular
Iterations	5	9	9
$\bar{x}$	$\begin{pmatrix} 1.035700 \\ -0.307235 \end{pmatrix}$	$\begin{pmatrix} 0.061991 \\ 0.123802 \end{pmatrix}$	$\begin{pmatrix} -0.791562 \\ -0.913038 \end{pmatrix}$
σ	$\begin{pmatrix} 3.064470 \\ 1.141072 \end{pmatrix}$	$\begin{pmatrix} 3.754477 \\ 1.630419 \end{pmatrix}$	$\begin{pmatrix} 3.254236 \\ 2.280980 \end{pmatrix}$
$\hat{w}$	$\begin{pmatrix} -0.577343 \\ -0.303446 \\ -0.119210 \end{pmatrix}$	$\begin{pmatrix} -6.402081 \\ 1.669199 \\ 1.370632 \end{pmatrix}$	$\begin{pmatrix} -7.169838 \\ 1.617820 \\ 2.055599 \end{pmatrix}$

(b) The equiprobability line defined by $p(y = 1 \mid x) = p(y = 0 \mid x) = \frac{1}{2}$ corresponds to the points x such that $\hat{w}^\intercal x = 0$, without forgetting to normalize and unnormalize. Therefore we can plot the data and the equiprobability line for the three datasets:

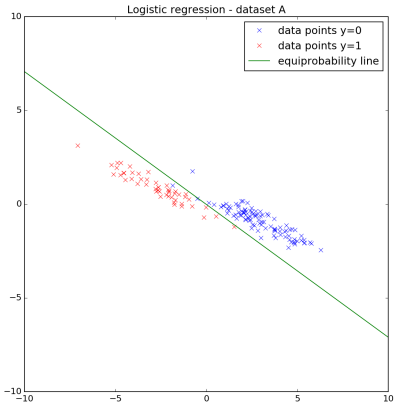

Figure 4: Logistic regression set A.

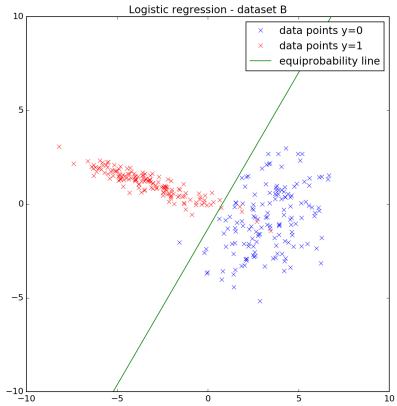

Figure 5: Logistic regression set B.

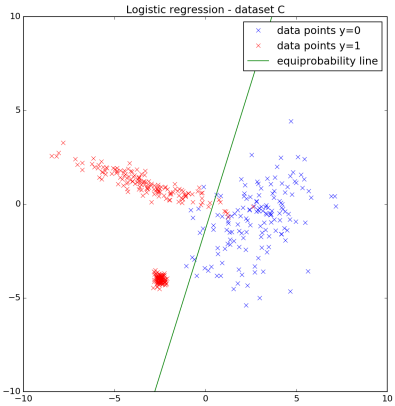

Figure 6: Logistic regression set C.

3 Linear regression

(a) The code for obtaining the numerical values of the parameters learnt can be found in the file ***linearregression.py***. Note that as the data $x^{(k)}_{1\leq k\leq n}$ was normalized into $\tilde{x}^{(k)}_{1\leq k\leq n}$ before running the algorithm thus we include the two parameters $\bar{x}$ and $\sigma \succeq 0$ ad defined before. Finally this gives us the numerical values for the

parameters $\hat{w}$ and $\hat{\sigma}$

Dataset	A	B	C
$\bar{x}$	$\begin{pmatrix} 1.035700 \\ -0.307235 \end{pmatrix}$	$\begin{pmatrix} 0.061991 \\ 0.123802 \end{pmatrix}$	$\begin{pmatrix} -0.791562 \\ -0.913038 \end{pmatrix}$
σ	$\begin{pmatrix} 3.064470 \\ 1.141072 \end{pmatrix}$	$\begin{pmatrix} 3.754477 \\ 1.630419 \end{pmatrix}$	$\begin{pmatrix} 3.254236 \\ 2.280980 \end{pmatrix}$
$\hat{w}$	$\begin{pmatrix} -0.809043 \\ -0.425155 \\ 0.333333 \end{pmatrix}$	$\begin{pmatrix} -0.391388 \\ 0.084441 \\ 0.500000 \end{pmatrix}$	$\begin{pmatrix} -0.415544 \\ -0.038780 \\ 0.625000 \end{pmatrix}$
$\hat{\sigma}$	0.199664	0.232947	0.249499

(b) The equiprobability line defined by $p(y = 1 \mid x) = p(y = 0 \mid x) = \frac{1}{2}$ corresponds to the points x such that $\hat{w}^\intercal x = \frac{1}{2}$, without forgetting to normalize and unnormalize. Therefore we can plot the data and the equiprobability line for the three datasets:

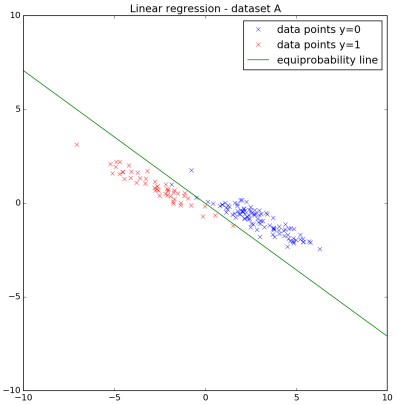

Figure 7: Linear regression set A.

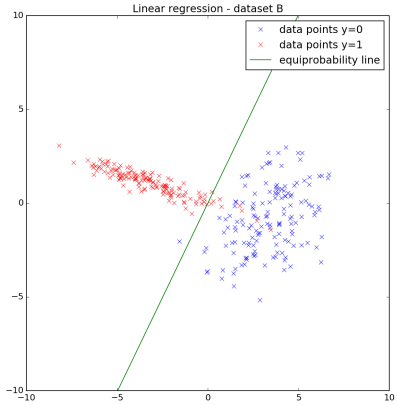

Figure 8: Linear regression set B.

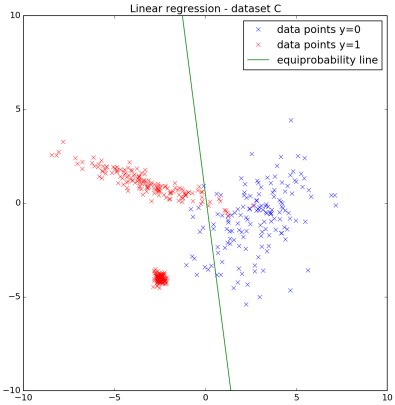

Figure 9: Linear regression set C.

4 Misclassification

(a) The misclassification results are summarized in the tables below plus a visualization is available.

Technique	Generative Model			Logistic Regression			Linear Regression		
Dataset	A	B	C	A	B	C	A	B	C
Train	2/150	9/300	24/400	2/150	6/300	16/400	2/150	9/300	22/400
Test	34/1500	83/2000	127/3000	31/1500	86/2000	68/3000	31/1500	83/2000	127/3000

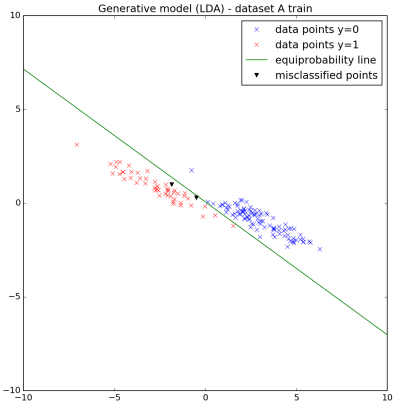

Figure 10: Generative model misclassification dataset A train.

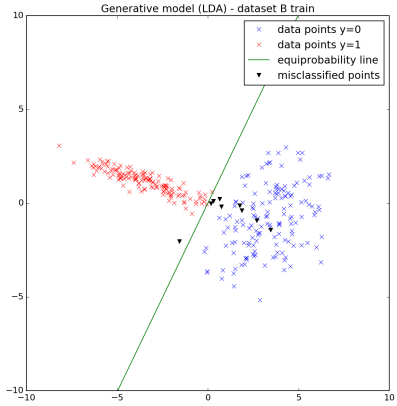

Figure 11: Generative model misclassification dataset B train.

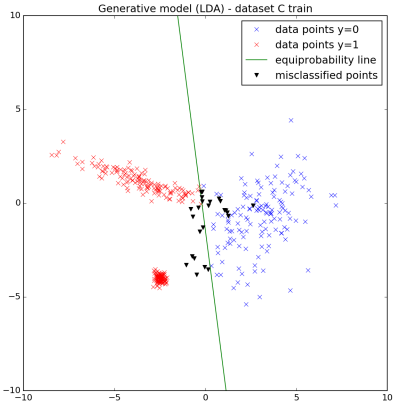

Figure 12: Generative model misclassification dataset C train.

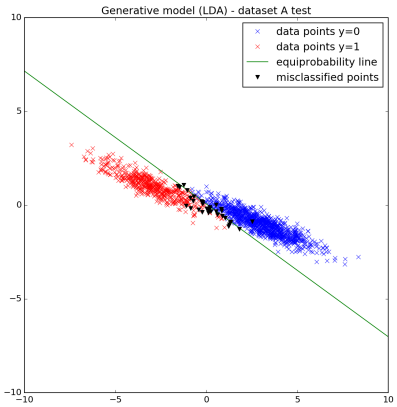

Figure 13: Generative model misclassification dataset A test.

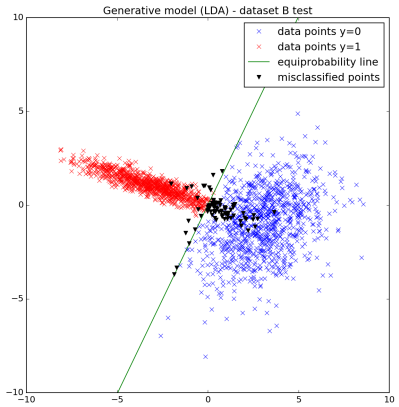

Figure 14: Generative model misclassification dataset B test.

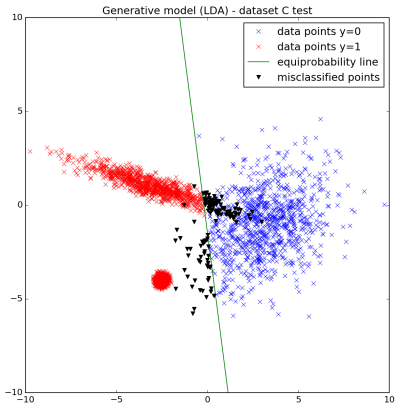

Figure 15: Generative model misclassification dataset C test.

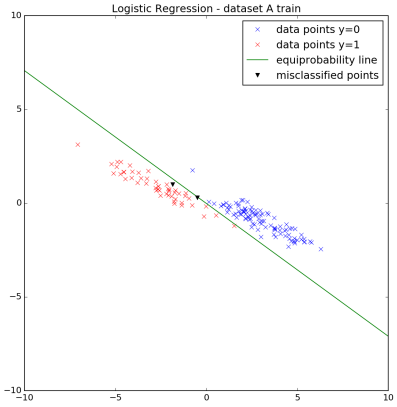

Figure 16: Logistic regression misclassification dataset A train.

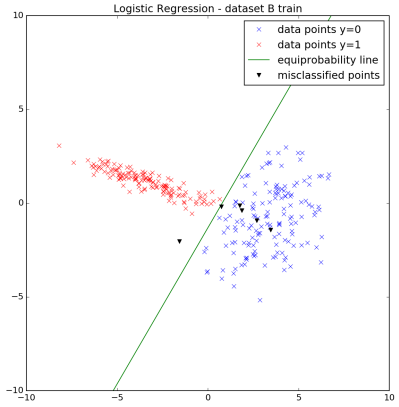

Figure 17: Logistic regression misclassification dataset B train.

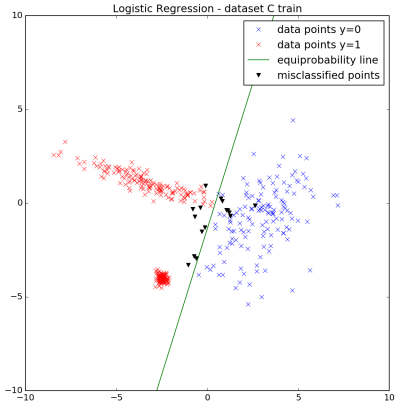

Figure 18: Logistic regression misclassification dataset C train.

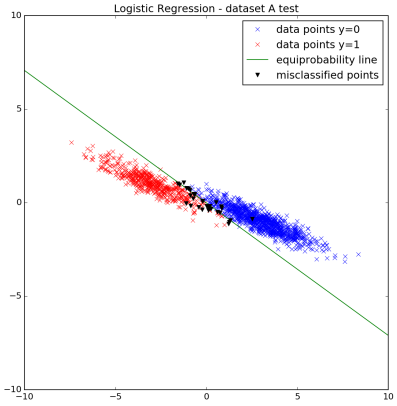

Figure 19: Logistic regression misclassification dataset A test.

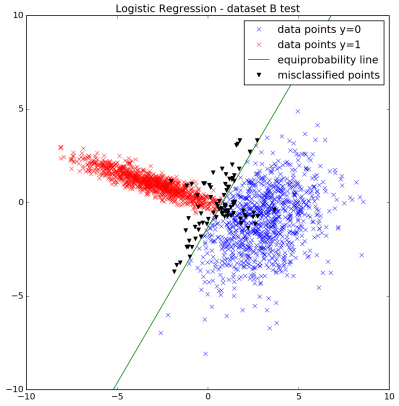

Figure 20: Logistic regression misclassification dataset B test.

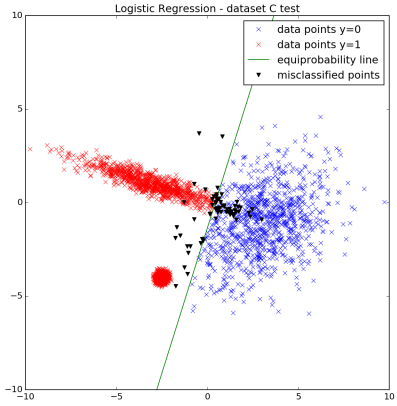

Figure 21: Logistic regression misclassification dataset C test.

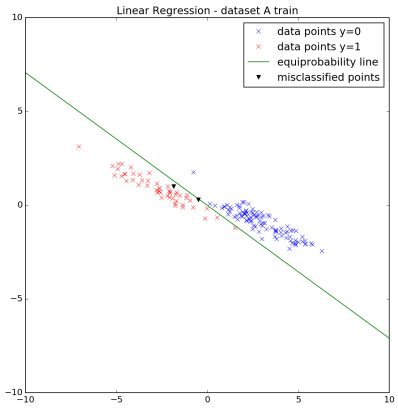

Figure 22: Linear regression misclassification dataset A train.

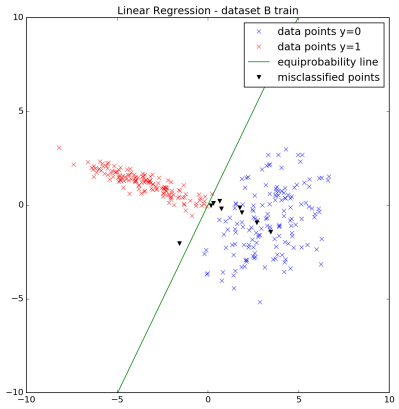

Figure 23: Linear regression misclassification dataset B train.

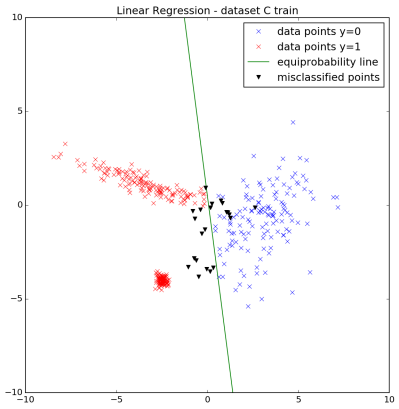

Figure 24: Linear regression misclassification dataset C train.

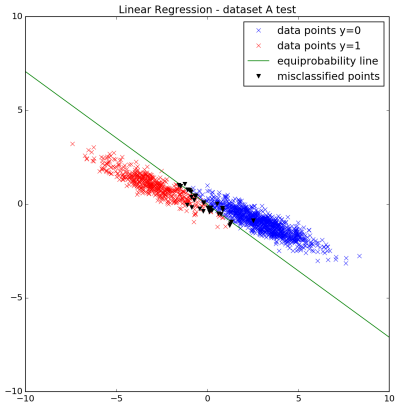

Figure 25: Linear regression misclassification dataset A test.

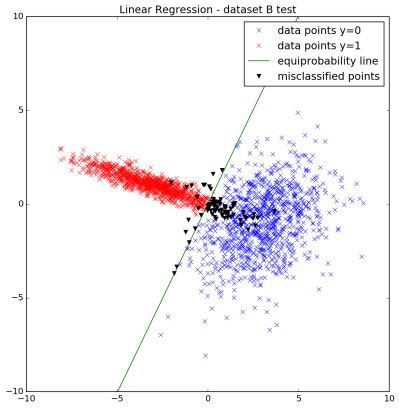

Figure 26: Linear regression misclassification dataset B test.

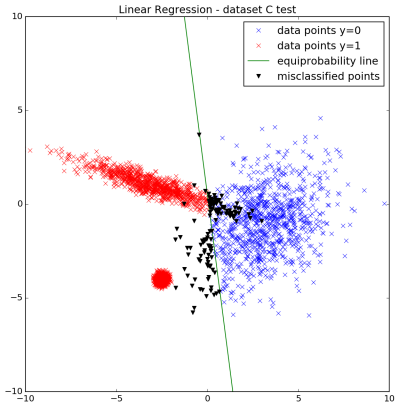

Figure 27: Linear regression misclassification dataset C test.

(b) We now examine more closely the results. We observe that all the algorithms have better performance on the train data than the test data, except for dataset C. This could be explained by the fact that in general, the algorithm is optimized to fit the train data and thus minimize the in-sample error (training set) and may not generalize to the out-of-sample error (testing set).
We can then ask ourselves why we have an odd behavior on the dataset C. One explanation might be that the dataset C is composed of 2 classes but actually is divided into 3 poles, thus a linear separation might not be the best approach leading to unpredictable generalization.
Finally all 3 techniques seem to have roughly the same results except for logistic regression on the dataset C which has en error improved by a factor of 2 compared to linear regression and generative model.

5 QDA model

(a) Using the notation from the LDA model section we have:

$$p_\pi(y = k) = \pi^k (1 - \pi)^{1-k}$$

$$p_{\mu_i,\Sigma_i}(x = z \mid y = i) = \frac{1}{\sqrt{(2\pi)^2 |\Sigma_i|}} \exp\left(-\frac{1}{2}(z - \mu_i)^\intercal \Sigma_i^{-1}(z - \mu_i)\right)$$

This change in covariance matrix doesn't modify the minimization for $\hat{\pi}$, $\hat{\mu}_0$ and $\hat{\mu}_1$ and thus:

$$\hat{\pi} = \frac{n}{N}$$

$$\hat{\mu}_0 = \frac{1}{N-n} \sum_{\substack{i=1 \\ y_i=0}}^{N} x_i$$

$$\hat{\mu}_1 = \frac{1}{n} \sum_{\substack{i=1 \\ y_i=1}}^{N} x_i$$

Now we need to minimize the two functions:

$$h_0 : \Sigma^{-1} \mapsto -(N-n) \log |\Sigma_0^{-1}| + \sum_{\substack{i=1 \\ y_i=0}}^{N} (x_i - \hat{\mu}_0)^\intercal \, \Sigma_0^{-1} \, (x_i - \hat{\mu}_0)$$

$$h_1 : \Sigma^{-1} \mapsto -n \log |\Sigma_1^{-1}| + \sum_{\substack{i=1 \\ y_i=1}}^{N} (x_i - \hat{\mu}_1)^\intercal \, \Sigma_1^{-1} \, (x_i - \hat{\mu}_1)$$

Which we already did in the previous section. Finally, to summarize:

$$\hat{\pi} = \frac{n}{N}$$

$$\hat{\mu}_0 = \frac{1}{N-n} \sum_{\substack{i=1 \\ y_i=0}}^{N} x_i$$

$$\hat{\mu}_1 = \frac{1}{n} \sum_{\substack{i=1 \\ y_i=1}}^{N} x_i$$

$$\hat{\Sigma}_0 = \frac{1}{N-n} \sum_{\substack{i=1 \\ y_i=0}}^{N} (x_i - \hat{\mu}_0) \, (x_i - \hat{\mu}_0)^\intercal$$

$$\hat{\Sigma}_1 = \frac{1}{n} \sum_{\substack{i=1 \\ y_i=1}}^{N} (x_i - \hat{\mu}_1) \, (x_i - \hat{\mu}_1)^\intercal$$

This yields the following numerical results using the file ***misclassification.py***.

Dataset	A	B	C
$\bar{x}$	$\begin{pmatrix} 1.035700 \\ -0.307235 \end{pmatrix}$	$\begin{pmatrix} 0.061991 \\ 0.123802 \end{pmatrix}$	$\begin{pmatrix} -0.791562 \\ -0.913038 \end{pmatrix}$
σ	$\begin{pmatrix} 3.064470 \\ 1.141072 \end{pmatrix}$	$\begin{pmatrix} 3.754477 \\ 1.630419 \end{pmatrix}$	$\begin{pmatrix} 3.254236 \\ 2.280980 \end{pmatrix}$
$\hat{\pi}$	0.333333	0.500000	0.625000
$\hat{\mu}_0$	$\begin{pmatrix} 0.608265 \\ -0.514112 \end{pmatrix}$	$\begin{pmatrix} 0.873277 \\ -0.588355 \end{pmatrix}$	$\begin{pmatrix} 1.101521 \\ 0.032728 \end{pmatrix}$
$\hat{\mu}_1$	$\begin{pmatrix} -1.216530 \\ 1.028224 \end{pmatrix}$	$\begin{pmatrix} -0.873277 \\ 0.588355 \end{pmatrix}$	$\begin{pmatrix} -0.660913 \\ -0.019637 \end{pmatrix}$
$\hat{\Sigma}_0$	$\begin{pmatrix} 0.246050 & -0.299557 \\ -0.299557 & 0.442215 \end{pmatrix}$	$\begin{pmatrix} 0.180111 & 0.173852 \\ 0.173852 & 1.113538 \end{pmatrix}$	$\begin{pmatrix} 0.273761 & 0.167835 \\ 0.167835 & 0.562143 \end{pmatrix}$
$\hat{\Sigma}_1$	$\begin{pmatrix} 0.287981 & -0.372014 \\ -0.372014 & 0.529702 \end{pmatrix}$	$\begin{pmatrix} 0.294664 & -0.218013 \\ -0.218013 & 0.194138 \end{pmatrix}$	$\begin{pmatrix} 0.270928 & -0.237371 \\ -0.237371 & 1.261686 \end{pmatrix}$

(b) To find the equiprobability line we write:

$$p(y = 1 \mid x) = p(y = 0 \mid x)$$

Thus we have:

$$\log|\hat{\Sigma}_0| + (x - \hat{\mu}_0)^\intercal \hat{\Sigma}_0^{-1} (x - \hat{\mu}_0) = \log|\hat{\Sigma}_1| + (x - \hat{\mu}_1)^\intercal \hat{\Sigma}_1^{-1} (x - \hat{\mu}_1)$$

Which yield the quadratic equation:

$$x^\intercal \left(\hat{\Sigma}_0^{-1} - \hat{\Sigma}_1^{-1}\right) x + 2\left(\hat{\mu}_1^\intercal \hat{\Sigma}_1^{-1} - \hat{\mu}_0^\intercal \hat{\Sigma}_0^{-1}\right) x + \hat{\mu}_0^\intercal \hat{\Sigma}_0^{-1} \hat{\mu}_0 - \hat{\mu}_1^\intercal \hat{\Sigma}_1^{-1} \hat{\mu}_1 + \log|\hat{\Sigma}_0| - \log|\hat{\Sigma}_1| = 0$$

Finally we can plot the data points and the line. The code for obtaining those figures can be found in the file ***qdamodel.py***.

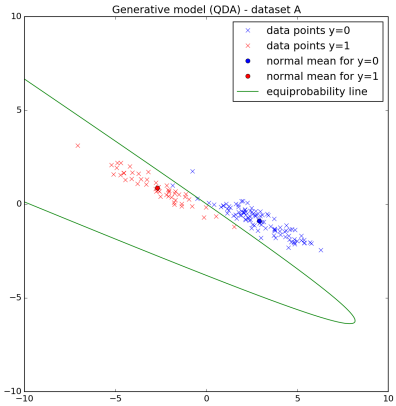

Figure 28: QDA model set A.

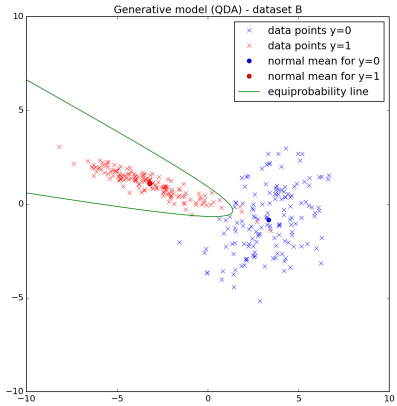

Figure 29: QDA model set B.

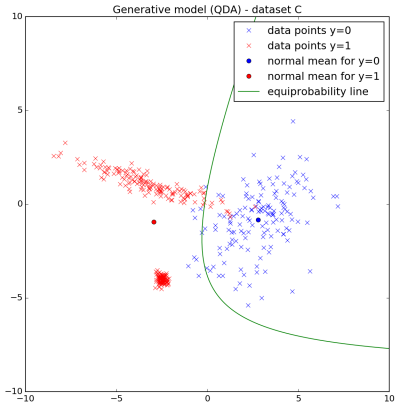

Figure 30: QDA model set C.

(c) As for the precedent classification techniques we compute the number of classified points and plot them on the figures below.

Technique	QDA Model		
Dataset	A	B	C
Train	2/150	4/300	18/400
Test	36/1500	40/2000	107/3000

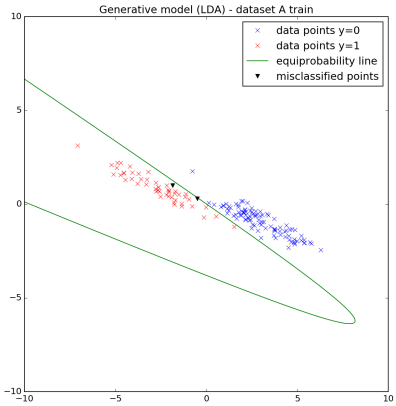

Figure 31: QDA model misclassification dataset A train.

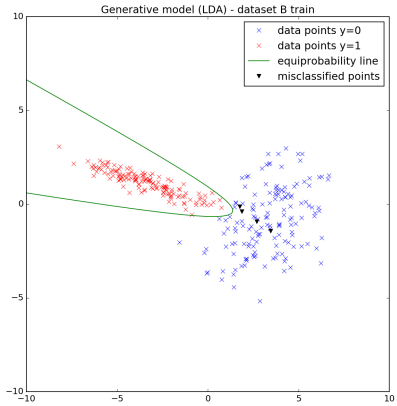

Figure 32: QDA model misclassification dataset B train.

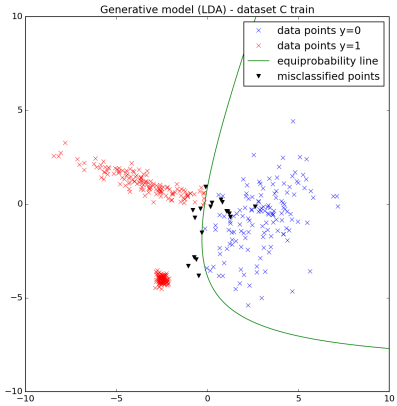

Figure 33: QDA model misclassification dataset C train.

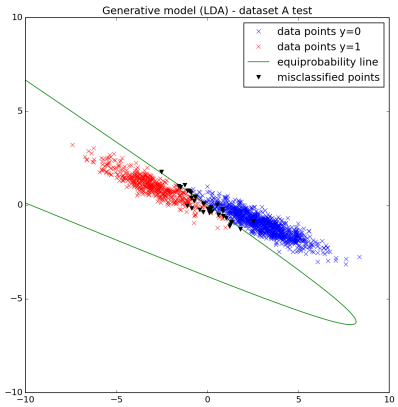

Figure 34: QDA model misclassification dataset A test.

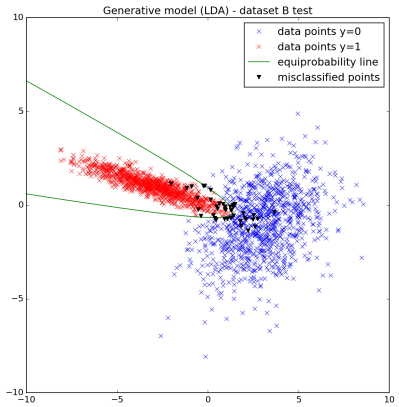

Figure 35: QDA model misclassification dataset B test.

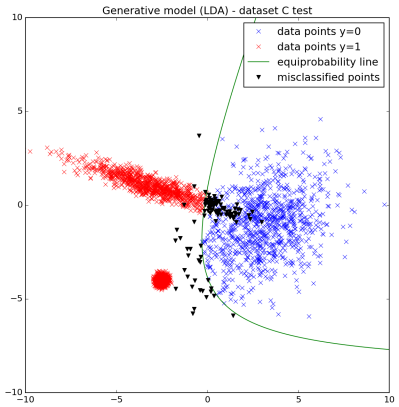

Figure 36: QDA model misclassification dataset C test.

(d) To help visualize the improvement of this technique we first plot the different equiprobability lines for the different datasets using ***equiprobabilitylines.py*** This explains the increase in performance on the dataset B as the QDA model is more suited to explain the red ellipsoid breakthrough into the blue circle.

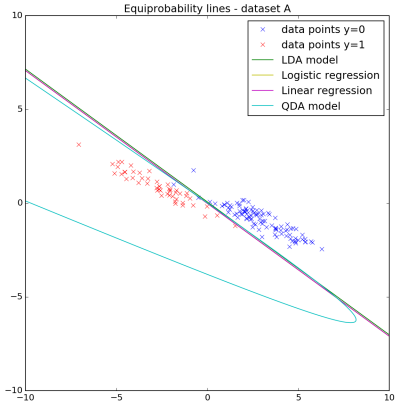

Figure 37: Equiprobability lines dataset A test.

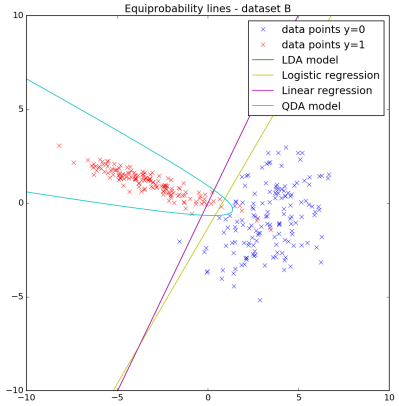

Figure 38: Equiprobability lines dataset B test.

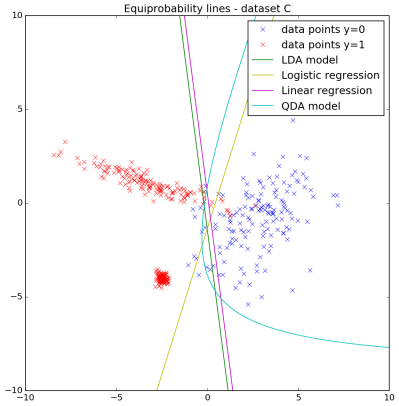

Figure 39: Equiprobability lines dataset C test.

Homework 2

Ayman YACHAOUI
Graphical Models
Master Data & Knowledge
Telecom ParisTech
ayman.yachaoui@telecom-paristech.fr

1 Entropy and Mutual Information

1. X discrete random variable on a finite space $\mathcal{X}$ with $|\mathcal{X}| = k$.
In the following we note $p(x) = P(X = x)$.

   (a) Thus we have the entropy $H(X)$:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

   As $0 \leq p(x) \leq 1$ then $-p(x) \log p(x) \geq 0$ because $f : x \mapsto -x \log x$ is non-negative on $[0, 1]$. Therefore:

$$H(X) \geq 0$$

   Besides the equality holds if and only if $p(x) \log p(x) = 0$ for all x, which implies that $p(x) = 0$ or $p(x) = 1$. It follows from $\sum_{x \in \mathcal{X}} p(x) = 1$ that there is only one x_c for which $p(x_c) = 1$ and $x \neq x_c \implies p(x) = 0$ which proves that $H(X) = 0$ only when X is constant.

   (b) The Kullback-Leibler divergence $D(p||q)$ is defined as:

$$\begin{aligned} D(p||q) &= \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \\ &= -\sum_{x \in \mathcal{X}} p(x) \log q(x) + \sum_{x \in \mathcal{X}} p(x) \log p(x) \\ &= \log k - H(X) \end{aligned}$$

   (c) Therefore as $D(p||q) \geq 0$ then:

$$H(X) \leq \log k$$

   Last but not least we prove that this is the tightest bound by observing that when $p = q$

$$H(X) = \log k$$

2. (X_1, X_2) pair of discrete random variable over the finite space $\mathcal{X}_1 \times \mathcal{X}_2$. Let $p_{1,2}, p_1$ and p_2 denote respectively the joint distribution, the marginal distribution of X_1 and the marginal distribution of X_2.

   (a) Thus we have the mutual information $I(X_1, X_2)$:

$$I(X_1, X_2) = \sum_{(x_1, x_2) \in \mathcal{X}_1 \times \mathcal{X}_2} p_{1,2}(x_1, x_2) \log \frac{p_{1,2}(x_1, x_2)}{p_1(x_1) p_2(x_2)}$$

   We observe that:

$$I(X_1, X_2) = D(p_{1,2}||p_1 p_2)$$

   Therefore $I(X_1, X_2) \geq 0$.

(b) Also we note that:

$$\begin{aligned} I(X_1, X_2) &= \sum_{(x_1,x_2)\in\mathcal{X}_1\times\mathcal{X}_2} p_{1,2}(x_1, x_2) \log \frac{p_{1,2}(x_1, x_2)}{p_1(x_1)p_2(x_2)} \\ &= \sum_{(x_1,x_2)\in\mathcal{X}_1\times\mathcal{X}_2} (p_{1,2}(x_1, x_2) (\log p_{1,2}(x_1, x_2) - \log p_1(x_1) - \log p_2(x_2))) \\ &= -H(X_1, X_2) - \sum_{(x_1,x_2)\in\mathcal{X}_1\times\mathcal{X}_2} p_{1,2}(x_1, x_2) (\log p_1(x_1) + \log p_2(x_2)) \end{aligned}$$

Besides:

$$\begin{aligned} \sum_{(x_1,x_2)\in\mathcal{X}_1\times\mathcal{X}_2} p_{1,2}(x_1, x_2) \log p_1(x_1) &= \sum_{x_1\in\mathcal{X}_1} \sum_{x_2\in\mathcal{X}_2} p_2(x_2)p_{1,2}(x_1 \mid x_2) \log p_1(x_1) \\ &= \sum_{x_1\in\mathcal{X}_1} \left(\log p_1(x_1) \left(\underbrace{\sum_{x_2\in\mathcal{X}_2} p_2(x_2)p_{1,2}(x_1 \mid x_2)}_{=p_1(x_1)} \right) \right) \\ &= \sum_{x_1\in\mathcal{X}_1} p_1(x_1) \log p_1(x_1) \\ &= -H(X_1) \end{aligned}$$

Therefore we have the final result:

$$I(X_1, X_2) = H(X_1) + H(X_2) - H(X_1, X_2)$$

(c) Using the precedent result we have:

$$H(X_1, X_2) = H(X_1) + H(X_2) - I(X_1, X_2)$$

Thus the joint distribution of maximal entropy is also the joint distribution of minimal mutual information which implies that $I(X_1, X_2) = D(p_{1,2}||p_1p_2) = 0$ and therefore $p_{1,2} = p_1p_2$ due to the Kullback-Leibler divergence nullity. This in turn implies that $p_1 \perp\!\!\!\perp p_2$.

2 Conditional Independence and Factorizations

1. Prove that $X \perp\!\!\!\perp Y \mid Z$ if and only if $p(x \mid y, z) = p(x \mid z)$ for all pairs (y, z) such that $p(y, z) > 0$ using only:

   (a) The definition of conditional independence: $p(a, b \mid c) = p(a \mid c)p(b \mid c) \quad p(c) > 0$

   (b) The definition of conditional probability: $p(a, b) = p(a \mid b)p(b)$

   (c) The summation rule: $\sum_a p(a \mid b) = 1$

   - If we have conditional independence $X \perp\!\!\!\perp Y \mid Z$, then:

$$\begin{aligned} p(x, y, z) &= p(x \mid y, z)p(y, z) && \text{(b) on } p(x, y, z) \\ &= p(x, y \mid z)p(z) && \text{(b) on } p(x, y, z) \\ &= p(x \mid z)p(z)p(y \mid z) && \text{(a) on } p(x, y \mid z) \\ &= p(x \mid z)p(y, z) && \text{(b) on } p(z)p(y \mid z) \end{aligned}$$

   And finally:

$$p(x \mid y, z)p(y, z) = p(x \mid z)p(y, z)$$

   Which implies that $p(x \mid y, z) = p(x \mid z)$ for all pairs (y, z) such that $p(y, z) > 0$.

- Now if $p(x \mid y, z) = p(x \mid z)$ for all pairs (y, z) such that $p(y, z) > 0$, then for all z such that $p(z) > 0$, $p(y, z) > 0$ and:

$$\begin{aligned} p(x, y, z) &= p(x, y \mid z)p(z) && \text{(b) on } p(x, y, z) \\ &= p(x \mid y, z)p(y, z) && \text{(b) on } p(x, y, z) \\ &= p(x \mid y, z)p(y \mid z)p(z) && \text{(b) on } p(y, z) \\ &= p(x \mid z)p(y \mid z)p(z) && \text{by hypothesis} \end{aligned}$$

And finally as $p(z) > 0$:

$$p(x, y \mid z) = p(x \mid z)p(y \mid z)$$

2. For any $p \in \mathcal{L}(G)$ using the directed graph model G we can write:

$$p(x, y, z, t) = p(x)p(y)p(z \mid x, y)p(t \mid z)$$

Besides, $X \perp\!\!\!\perp Y \mid Z$ is equivalent to $p(x, y \mid z) = p(x \mid z)p(y \mid z)$ for all z such that $p(z) > 0$. Thus:

$$\begin{aligned} p(x, y \mid z) &= \frac{\sum_t p(x, y, z, t)}{p(z)} \\ &= \frac{p(x)p(y)p(z \mid x, y) \sum_t p(t \mid z)}{p(z)} \\ &= p(x \mid z)p(y \mid z)p(z)p(z \mid x, y) \end{aligned}$$

Therefore by picking a (x, y, z) such that $p(x, y \mid z) \neq 0$ and $0 < p(z) < 1$ we have $p(x, y \mid z) \neq p(x \mid z)p(y \mid z)$. Which disproves the proposition.
For instance if X and Y correspond to the value of two dices and Z is their sum then, knowing Z, X and Y are not at all independent as setting one means setting the other. We can illustrate this with an example:

$$\begin{aligned} p(x = 6, y = 6 \mid z = 11) &= 0 \\ p(x = 6 \mid z = 11) = p(y = 6 \mid z = 11) &= 0.5 \end{aligned}$$

3. If $X \perp\!\!\!\perp Y \mid Z$ and $X \perp\!\!\!\perp Y$ then for all z such that $p(z) > 0$ we have:

$$\begin{aligned} p(x, y) &= p(x)p(y) \\ p(x, y \mid z) &= p(x \mid z)p(y \mid z) \end{aligned}$$

(a) Thus if Z is a binary variable we can write:

$$\begin{aligned} p(x, y) &= p(x)p(y) \\ &= \sum_z p(x, y, z) \\ &= \sum_z p(x, y \mid z)p(z) \\ &= \sum_z p(x \mid z)p(y \mid z)p(z) \\ &= \sum_z p(x \mid z)p(y, z) \\ &= \sum_z p(x \mid z)p(z \mid y)p(y) \\ &= p(y) \left(p(x \mid z_0)p(z_0 \mid y) + p(x \mid z_1)p(z_1 \mid y)\right) \end{aligned}$$

Therefore if $p(y) \neq 0$

$$p(x) = p(x \mid z_0)p(z_0 \mid y) + p(x \mid z_1)p(z_1 \mid y)$$

- If there is just one y such that $p(y) \neq 0$ then Y is constant and thus $Y \perp\!\!\!\perp Z$.
- Otherwise we denote y' a different point such that $p(y') \neq 0$, and we have:

$$\begin{aligned} p(x) &= p(x \mid z_0)p(z_0 \mid y) + p(x \mid z_1)p(z_1 \mid y) \\ &= p(x \mid z_0)p(z_0 \mid y') + p(x \mid z_1)p(z_1 \mid y') \end{aligned}$$

Which yields for all x:

$$p(x \mid z_0)(p(z_0 \mid y) - p(z_0 \mid y')) = -p(x \mid z_1)(p(z_1 \mid y) - p(z_1 \mid y'))$$

– If either $p(z_0 \mid y) - p(z_0 \mid y') \neq 0$ or $p(z_1 \mid y) - p(z_1 \mid y') \neq 0$ we consider the first case without loss of generality then:

$$\begin{aligned} 0 \leq p(x \mid z_0) &= -p(x \mid z_1)\frac{p(z_1 \mid y) - p(z_1 \mid y')}{p(z_0 \mid y) - p(z_0 \mid y')} \\ &= -p(x \mid z_1)\frac{p(z_1)p(y') - p(z_1)p(y)}{p(z_0)p(y') - p(z_0)p(y)} \\ &= -\frac{p(x)}{p(z_0)} \leq 0 \end{aligned}$$

Therefore $p(x \mid z_0) = p(x) = 0$ and $p(x \mid z_1) = p(x) - p(x \mid z_0) = 0$ which implies that: $p(x \mid z_0) = p(x \mid z_1)$ and $X \perp\!\!\!\perp Z$.

– Otherwise we have:

$$\begin{aligned} p(z_0 \mid y) &= p(z_0 \mid y') \\ p(z_1 \mid y) &= p(z_1 \mid y') \end{aligned}$$

Therefore $p(z \mid y) = p(z \mid y')$. This means that for all y such that $p(y) > 0$ then $p(z \mid y)$ doesn't depend on y. Besides when $p(y) = 0$ then $p(y, z) = 0 = p(y)p(z)$. This ends the proof and $Y \perp\!\!\!\perp Z$.

(b) In general the statement is false. For instance if we X and Y are two independent random integers between 1 and 6 (dices), then we chose $Z = (X, Y)$.
By definition $X \perp\!\!\!\perp Y$. Also if $z \neq (x, y)$, $p(x, y \mid z) = 0 = p(x \mid z)p(y \mid z)$ because either $p(x \mid z) = 0$ or $p(y \mid z) = 0$. Otherwise $z = (x, y)$ and $p(x, y \mid z) = 1 = p(x \mid z)p(y \mid z)$. Which proves that $X \perp\!\!\!\perp Y \mid Z$.
But X and Y are completely determined by Z which means that they are not independent from Z.

3 Distributions factorizing in a graph

1. $G = (V, E)$ is a directed graph thus $\mathcal{L}(G)$ is the family of functions $\{f_k(x_k, x_{\pi_k}) : k \in \mathcal{V}\}$ such that $f_k(x_k, x_{\pi_k}) \geq 0$ and $\sum_{x_k} f_k(x_k, x_{\pi_k}) = 1$

$$\begin{aligned} p(x_1, x_2, ..., x_n) &= \prod_{k=1}^{n} f_k(x_k, x_{\pi_k}) \\ &= f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_j}) \prod_{\substack{k=1 \\ k \neq i \\ k \neq j}}^{n} f_k(x_k, x_{\pi_k}) \\ &= f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i) \prod_{\substack{k=1 \\ k \neq i \\ k \neq j}}^{n} f_k(x_k, x_{\pi_k}) \end{aligned}$$

In the new graph G', $\pi'_j = \pi_i$, $\pi'_i = \pi_i \cup \{j\}$ and $\pi'_k = \pi_k$ otherwise thus $\mathcal{L}(G')$ is the family of functions $\{h_k(x_k, x_{\pi'_k}) : k \in \mathcal{V}\}$ such that $h_k(x_k, x_{\pi'_k}) \geq 0$ and $\sum_{x_k} h_k(x_k, x_{\pi'_k}) = 1$ and also:

$$\begin{aligned} p(x_1, x_2, ..., x_n) &= \prod_{k=1}^{n} h_k(x_k, x_{\pi'_k}) \\ &= h_i(x_i, x_{\pi'_i}) h_j(x_j, x_{\pi'_j}) \prod_{\substack{k=1 \\ k\neq i \\ k\neq j}}^{n} h_k(x_k, x_{\pi_k}) \\ &= h_i(x_i, x_{\pi_i}, x_j) h_j(x_j, x_{\pi_i}) \prod_{\substack{k=1 \\ k\neq i \\ k\neq j}}^{n} h_k(x_k, x_{\pi_k}) \end{aligned}$$

Thus we chose:

$$h_j(x_j, x_{\pi_i}) = \sum_{x_i} f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i) \geq 0$$

$$h_i(x_i, x_{\pi_i}, x_j) = \frac{f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i)}{\sum_{x_i} f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i)} \geq 0$$

And we just need to verify that

$$\begin{aligned} \sum_{x_j} h_j(x_j, x_{\pi'_j}) = \sum_{x_j} h_j(x_j, x_{\pi_i}) &= \sum_{x_j} \sum_{x_i} f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i) \\ &= \sum_{x_i} f_i(x_i, x_{\pi_i}) \sum_{x_j} f_j(x_j, x_{\pi_j}) \\ &= \sum_{x_i} f_i(x_i, x_{\pi_i}) \\ &= 1 \end{aligned}$$

$$\begin{aligned} \sum_{x_i} h_i(x_i, x_{\pi'_i}) = \sum_{x_i} h_i(x_i, x_{\pi_i}, x_j) &= \sum_{x_i} \frac{f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i)}{\sum_{x_i} f_i(x_i, x_{\pi_i}) f_j(x_j, x_{\pi_i}, x_i)} \\ &= 1 \end{aligned}$$

Therefore we have proved that $\mathcal{L}(G) \subset \mathcal{L}(G')$.
Using the exact same method with the functions $f_i(x_i, x_{\pi_i})$ and $f_j(x_j, x_{\pi_i}, x_i)$ defined below we also prove that $\mathcal{L}(G') \subset \mathcal{L}(G)$.

$$f_i(x_i, x_{\pi_i}) = \sum_{x_j} h_i(x_i, x_{\pi_i}, x_j) h_j(x_j, x_{\pi_i}) \geq 0$$

$$f_j(x_j, x_{\pi_i}, x_i) = \frac{h_i(x_i, x_{\pi_i}, x_j) h_j(x_j, x_{\pi_i})}{\sum_{x_j} h_i(x_i, x_{\pi_i}, x_j) h_j(x_j, x_{\pi_i})} \geq 0$$

Finally we can conclude that $\mathcal{L}(G) = \mathcal{L}(G')$.

2. G is a directed tree, thus $p \in \mathcal{L}(G)$ if:

$$p(x_1, x_2, ..., x_n) = \prod_{k=1}^{n} f_k(x_k, x_{\pi_k})$$

$$\sum_{x_k} f_k(x_k, x_{\pi_k}) = 1$$

$$f_k(x_k, x_{\pi_k}) \geq 0$$

G' is the corresponding undirected tree, thus noting $\mathcal{C}$ the set of maximal cliques $p' \in \mathcal{L}(G')$ if:

$$p'(x_1, x_2, ..., x_n) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_{\mathcal{C}}(x_{\mathcal{C}})$$
$$Z = \sum_x \prod_{C \in \mathcal{C}} \psi_{\mathcal{C}}(x_{\mathcal{C}})$$
$$\psi_{\mathcal{C}}(x_{\mathcal{C}}) \geq 0$$

As G is a directed tree then considering a topological ordering of the vertices, for all $k > 1$ every pair (x_k, x_{π_k}) is a clique of two vertices and by choosing $\mathcal{C}_{tree} = \{C_k = (k, \pi_k) \mid k > 1\}$ then:

$$p(x_1, x_2, ..., x_n) = f_1(x_1) f_2(x_2, x_1) \prod_{k>2}^{n} f_k(x_k, x_{\pi_k})$$

Thus by choosing:

$$\psi'_k(x_k, x_{\pi_k}) = f_k(x_k, x_{\pi_k}) \geq 0, \quad \forall k > 2$$
$$\psi'_2(x_2, x_1) = f_1(x_1) f_2(x_2, x_1) \geq 0$$

We obtain:

$$p(x_1, x_2, ..., x_n) = \prod_{k>1}^{n} \psi'_k(x_k, x_{\pi_k}) = \prod_{k>1}^{n} \psi'_{C_k}(x_{C_k}) = \prod_{C \in \mathcal{C}_{tree}} \psi'_C(x_C)$$

And

$$Z = \sum_x \prod_{C \in \mathcal{C}_{tree}} \psi'_{\mathcal{C}}(x_{\mathcal{C}}) = \sum_x \prod_{k>1}^{n} \psi'_k(x_k, x_{\pi_k}) = \sum_x \prod_{k=1}^{n} f_k(x_k, x_{\pi_k}) = 1$$

Thus $\mathcal{L}(G) \subset \mathcal{L}(G')$.

We now use the same decomposition of the $\psi_C(x_C)$ and by choosing:

$$f_k(x_k, x_{\pi_k}) = \frac{\psi_k(x_k, x_{\pi_k})}{\sum_{x_k} \psi_k(x_k, x_{\pi_k})} \geq 0, \quad \forall k > 1$$
$$f_1(x_1) = \frac{\sum_{x_2} \psi_2(x_2, x_1)}{\sum_{x_1, x_2} \psi_2(x_2, x_1)} \geq 0$$

We verify that:

$$\sum_{x_k} f_k(x_k, x_{\pi_k}) = \sum_{x_k} \frac{\psi_k(x_k, x_{\pi_k})}{\sum_{x_k} \psi_k(x_k, x_{\pi_k})} = 1$$
$$\sum_{x_1} f_1(x_1) = \sum_{x_1} \frac{\sum_{x_2} \psi_2(x_2, x_1)}{\sum_{x_1, x_2} \psi_2(x_2, x_1)} = 1$$

And finally we have the decomposition:

$$
\begin{aligned}
p'(x_1, x_2, ..., x_n) &= \frac{\prod_{k>1}^{n} \psi_k(x_k, x_{\pi_k})}{\sum_x \prod_{k>1}^{n} \psi_k(x_k, x_{\pi_k})} \\
&= \frac{\prod_{k>1}^{n} \left(f_k(x_k, x_{\pi_k}) \sum_{x_k} \psi_k(x_k, x_{\pi_k}) \right)}{\sum_x \prod_{k>1}^{n} \psi_k(x_k, x_{\pi_k})} \\
&= \left(\prod_{k>1}^{n} f_k(x_k, x_{\pi_k}) \right) \frac{\prod_{k>1}^{n} \sum_{x_k} \psi_k(x_k, x_{\pi_k})}{\sum_x \prod_{k>1}^{n} \psi_k(x_k, x_{\pi_k})} \\
&= \left(\prod_{k>1}^{n} f_k(x_k, x_{\pi_k}) \right) \frac{\sum_{x_2} \psi_2(x_2, x_{\pi_2})}{\sum_{x_1, x_2} \psi_2(x_2, x_{\pi_2})} \\
&= \left(\prod_{k>1}^{n} f_k(x_k, x_{\pi_k}) \right) \frac{\sum_{x_2} \psi_2(x_2, x_1)}{\sum_{x_1, x_2} \psi_2(x_2, x_1)} \\
&= \left(\prod_{k>1}^{n} f_k(x_k, x_{\pi_k}) \right) f_1(x_1) \\
&= \prod_{k=1}^{n} f_k(x_k, x_{\pi_k})
\end{aligned}
$$

Therefore $\mathcal{L}(G') \subset \mathcal{L}(G)$ which ends the proof as we can now conclude that $\mathcal{L}(G') = \mathcal{L}(G)$.

4 Implementation - Gaussian mixtures

(a) As can be observed on the figures below, the K-means algorithm seem to converge on the same distorsion value but produces different clustering. This is due to the fact that every minimum is a local minimum because every permutation yields a new local minimum. We also include a figure representing an extreme case. Last but not least we include a plot of the distorsion for different random initialization. It can be observed that making a few random initialization ensure that we find a good solution.

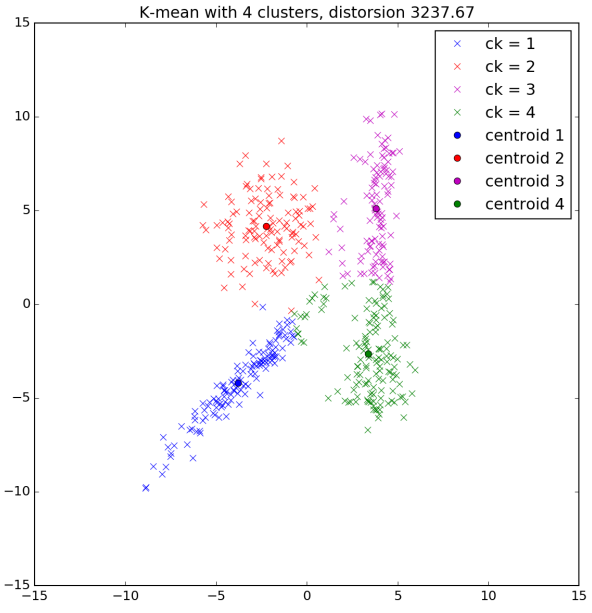

Figure 1: K-means clustering regular case 1.

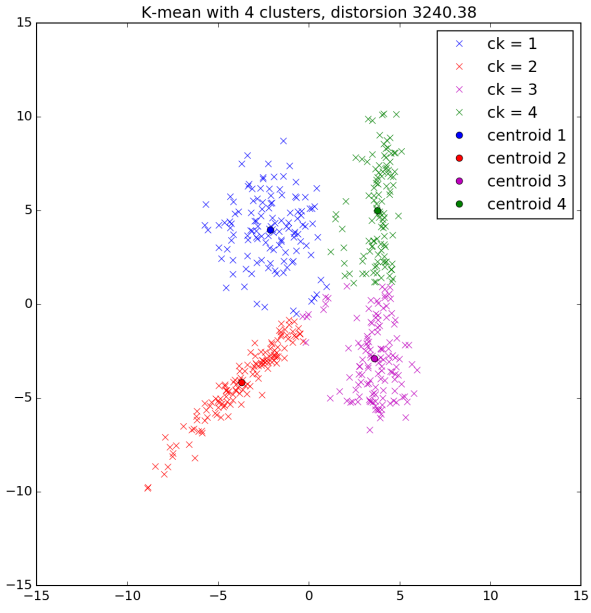

Figure 2: K-means clustering regular case 2.

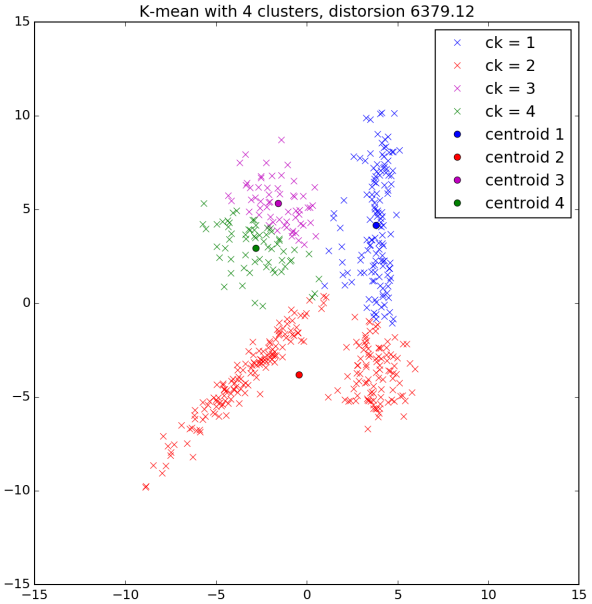

Figure 3: K-means clustering extreme case.

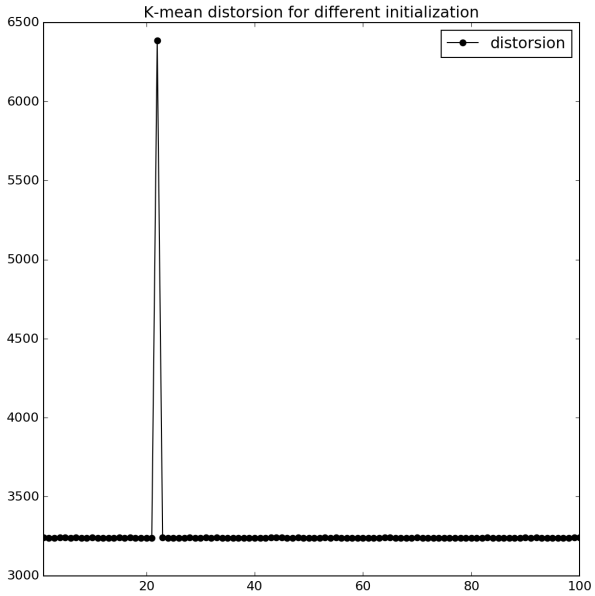

Figure 4: K-means distorsion.

(b) We focus on the maximization step thus:

$$\left(\pi_k^{(t)}, \mu_k^{(t)}, \Sigma_k^{(t)}\right)_{1\leq k\leq K} = \underset{\theta=(\pi_k,\mu_k,\Sigma_k)_{1\leq k\leq K}}{\arg\max} \mathbb{E}_{q^{(t)}}\left[\tilde{\ell}\left(\theta\right)\right]$$

With:

$$\Sigma_k^{(t)} = \sigma_k^{(t)} I_2 \succ 0$$

$$\mathbb{E}_{q^{(t)}}\left[\tilde{\ell}\left(\theta\right)\right] = \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)}\left(\log\pi_k + \log\mathcal{N}\left(x_i, \mu_k, \Sigma_k\right)\right)$$

- First we minimize $g : \pi \mapsto -\sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\pi_k$ subject to $\mathbf{1}^\intercal\pi = 1$ and denoting the Laplacian multiplier λ and taking the Laplacian derivative we find for all k:

$$\hat{\pi}_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)}}{\lambda}$$

Thus by reinjecting in the equality constraint $\mathbf{1}^\intercal\pi = 1$ we obtain:

$$\hat{\pi}_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)}}{\sum_{j=1}^{K}\sum_{i=1}^{n} q_{ij}^{(t)}}$$

- Now we consider $h : (\mu, \Sigma) \mapsto \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\mathcal{N}\left(x_i, \mu_k, \Sigma_k\right)$, then:

$$\begin{aligned}
h : (\mu, \Sigma) &= \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\mathcal{N}\left(x_i, \mu_k, \Sigma_k\right) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\left(\frac{1}{\sqrt{(2\pi)^2|\Sigma_k|}}\exp\left(-\frac{1}{2}(x_i-\mu_k)^\intercal\Sigma_k^{-1}(x_i-\mu_k)\right)\right) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\left(\frac{1}{\sqrt{(2\pi)^2|\sigma_k I_2|}}\exp\left(-\frac{1}{2}(x_i-\mu_k)^\intercal\left(\sigma_k I_2\right)^{-1}(x_i-\mu_k)\right)\right) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} q_{ik}^{(t)} \log\left(\frac{1}{2\pi\sigma_k}\exp\left(-\frac{1}{2\sigma_k}(x_i-\mu_k)^\intercal(x_i-\mu_k)\right)\right) \\
&= \sum_{k=1}^{K}\sum_{i=1}^{n} q_{ik}^{(t)}\left(-\log 2\pi - \log\sigma_k - \frac{\|x_i-\mu_k\|_2^2}{2\sigma_k}\right)
\end{aligned}$$

– Observing $h_\mu : \mu \mapsto h(\mu, \Sigma)$ is a concave function we find the points where its gradient is nul thus for all k:

$$\frac{\partial h_\mu}{\partial \mu_k}(\hat{\mu}_k) = 0 = \sum_{i=1}^{n} q_{ik}^{(t)} \frac{x_i - \mu_k}{\sigma_k}$$

Which yields:

$$\mu_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)} x_i}{\sum_{i=1}^{n} q_{ik}^{(t)}}$$

– Last but not least we consider $h_\sigma : \mu \mapsto h(\mu, \sigma)$ by substituting σ for Σ. Then:

$$\begin{aligned}\frac{\partial h_\sigma}{\partial \sigma_k} &= \sum_{i=1}^{n} q_{ik}^{(t)} \left(-\frac{1}{\sigma_k} + \frac{\|x_i - \mu_k\|_2^2}{2\sigma_k^2} \right) \\ &= \frac{1}{\sigma_k} \left(\frac{\sum_{i=1}^{n} q_{ik}^{(t)} \|x_i - \mu_k\|_2^2}{2\sigma_k} - \sum_{i=1}^{n} q_{ik}^{(t)} \right)\end{aligned}$$

We note:

$$\hat{\sigma}_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)} \|x_i - \mu_k\|_2^2}{2 \sum_{i=1}^{n} q_{ik}^{(t)}}$$

Then h_σ is increasing on σ_k when $\sigma_k \leq \hat{\sigma}_k$ and decreasing when $\sigma_k \geq \hat{\sigma}_k$ which ensure that the maximum is reach for $\hat{\sigma}_k$.

We now have all the parameters of the maximization step which are summarized below:

$$\begin{cases} \hat{\pi}_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)}}{\sum_{j=1}^{K} \sum_{i=1}^{n} q_{ij}^{(t)}} \\ \mu_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)} x_i}{\sum_{i=1}^{n} q_{ik}^{(t)}} \\ \hat{\sigma}_k = \frac{\sum_{i=1}^{n} q_{ik}^{(t)} \|x_i - \mu_k\|_2^2}{2 \sum_{i=1}^{n} q_{ik}^{(t)}} \end{cases}$$

Finally we can train the EM algorithm on the training data (left figure) and visualize the results with the test data (right figure). On the following plot we have represented the Gaussian mixture centers $(\mu_k)_{1 \leq k \leq K}$ with colored circles, the $(\sigma_k)_{1 \leq k \leq K}$ by drawing the ellipses corresponding to 90% of the mass and finally the latent variable $(z_i)_{1 \leq i \leq n}$ by coloring the datapoints.

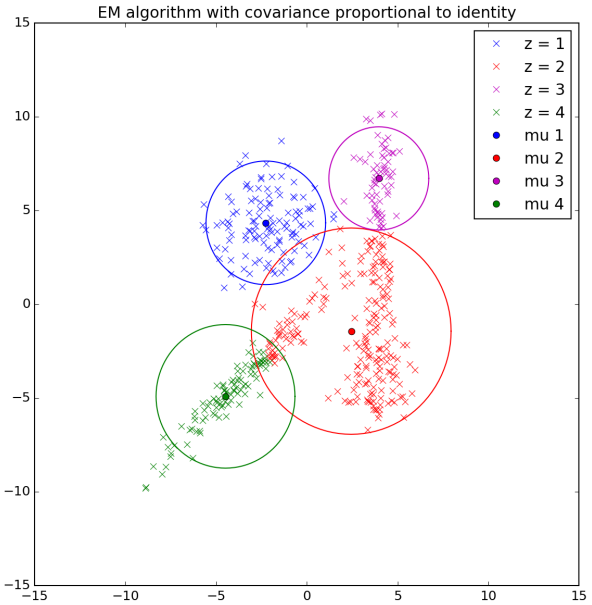

Figure 5: EM algorithm training with covariance matrices proportional to identity.

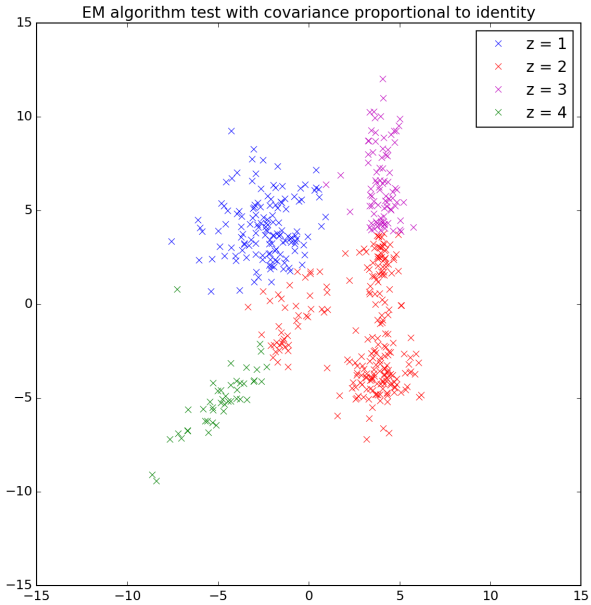

Figure 6: EM algorithm testing with covariance matrices proportional to identity.

(c) We now use general covariance matrices $(\Sigma_k)_{1\leq k\leq K}$ and use the maximization step:

$$\begin{cases} \hat{\pi}_k = \frac{\sum_{i=1}^n q_{ik}^{(t)}}{\sum_{j=1}^K \sum_{i=1}^n q_{ij}^{(t)}} \\ \mu_k = \frac{\sum_{i=1}^n q_{ik}^{(t)} x_i}{\sum_{i=1}^n q_{ik}^{(t)}} \\ \hat{\Sigma}_k = \frac{\sum_{i=1}^n q_{ik}^{(t)} (x_i - \mu_k)(x_i - \mu_k)^\intercal}{\sum_{i=1}^n q_{ik}^{(t)}} \end{cases}$$

This yields the classification below where we use the same rules as above to represent graphically the Gaussian centers $(\mu_k)_{1\leq k\leq K}$, the covariance matrices $(\Sigma_k)_{1\leq k\leq K}$ and the latent variables $(z_i)_{1\leq i\leq n}$. The image on the left corresponds to the training and the testing is represented on the right.

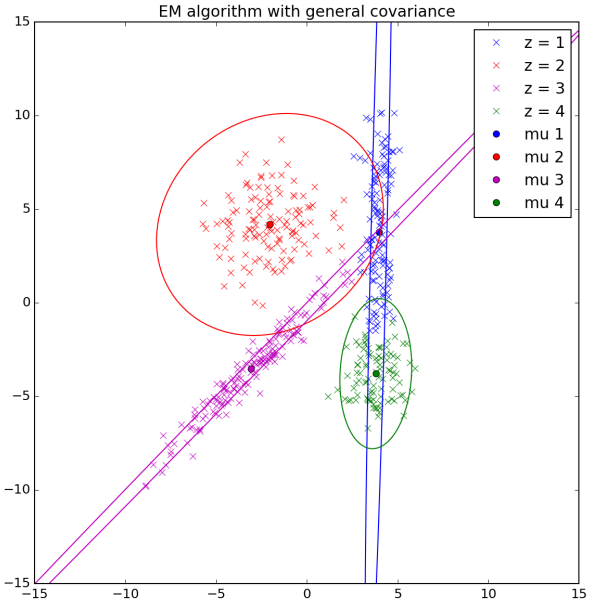

Figure 7: EM algorithm training with general covariance matrices.

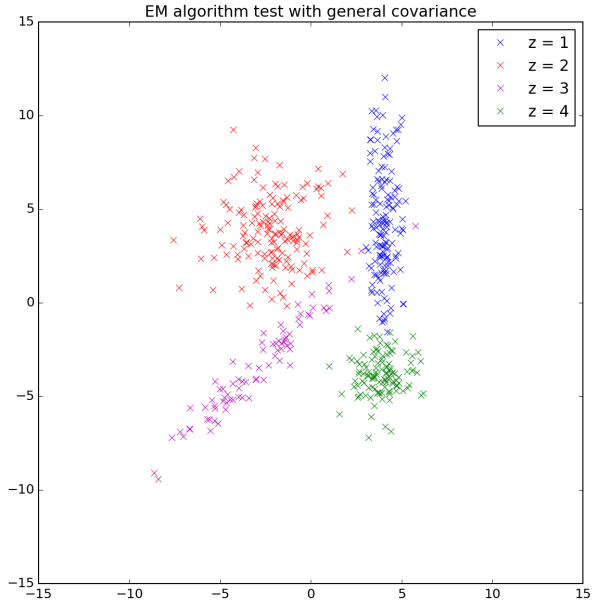

Figure 8: EM algorithm testing with general covariance matrices.

(d) Looking at the data classification for the two type of EM algorithm we clearly observed that constraining the covariance matrices to be proportional to the identity might have been a bit too much. Even the k-mean seemed to provide better results. On the other hand when using general covariance matrices, the classification seems to improve dramatically. This is just a qualitative feeling.

To get a more quantitative approach, we compute the marignal log-likelihoods of the two EM algorithms. The results are summarized in the table below.

Marginal log-likelihood		
Algorithm type	**Training dataset**	**Test dataset**
Identity covariance	-2646	-2694
General	-2328	-2409

From this we can draw two conclusion. First both algorithm have better log-likelihoods on the training dataset, which is to be expected as they are biased toward fitting the training dataset. And second, the general EM algorithm sees an increase in log-likelihood of approximately 300 over the simpler EM algorithm. This demonstrate the intuition that we had that the general EM algorithm gives a way more plausible solution than the simplified version.

Homework 3

Ayman YACHAOUI
Graphical Models
Master Data & Knowledge
Telecom ParisTech
ayman.yachaoui@telecom-paristech.fr

April 1, 2017

1 α, β,γ and ξ recursions

The recursion α, β, γ and ξ are defined as follow:

$$\alpha_t(q_t) \triangleq p(q_t, u_0, \dots, u_t)$$
$$\beta_t(q_t) \triangleq p(u_{t+1}, \dots, u_T \mid q_t)$$
$$\gamma_t(q_t) \triangleq p(q_t \mid u_1, ..., u_T)$$
$$\xi_t(q_t, q_{t+1}) \triangleq p(q_t, q_{t+1} \mid u_1, ..., u_T)$$

And they can be computed recursively using:

$$\alpha_1(q_1) = p(u_1 \mid q_1)p(q_1)$$
$$\alpha_{t+1}(q_{t+1}) = p(u_{t+1} \mid q_{t+1}) \sum_{q_t} p(q_{t+1} \mid q_t)\alpha_t(q_t)$$
$$\beta_T(q_T) = 1$$
$$\beta_t(q_t) = \sum_{q_{t+1}} p(q_{t+1} \mid q_t)p(u_{t+1} \mid q_{t+1})\beta_{t+1}(q_{t+1})$$

Additionally we have:

$$p(q_{t+1} = j \mid q_t = i) = a_{ij}$$
$$p(u_t \mid q_t) = \frac{1}{2\pi\sqrt{|\Sigma_{q_t}|}} \exp\left(-\frac{1}{2}(u_t - \mu_{q_t})^\intercal \Sigma_{q_t}^{-1}(u_t - \mu_{q_t})\right)$$

Which enables us to compute recursively the γ recursion:

$$\gamma_t(q_t) = \frac{\alpha_t(q_t)\beta_t(q_t)}{\sum_q \alpha_t(q)\beta_t(q)}$$

And the ξ recursion:

$$\xi_t(q_t, q_{t+1}) = \frac{\alpha_t(q_t)\gamma_{t+1}(q_{t+1})p(u_{t+1} \mid q_{t+1})p(q_{t+1} \mid q_t)}{\alpha_{t+1}(q_{t+1})}$$

We take care of numerical errors by using the identity:

$$\log \sum_i x_i = \max_i \log(x_i) + \log \sum_i \exp(\log(x_i) - \max_i \log(x_i))$$

2 Plot γ for the training dataset

We can thus plot the probability $\gamma_t(q_t) = p(q_t \mid u_1, ..., u_T)$ for the first 100 points of the training dataset:

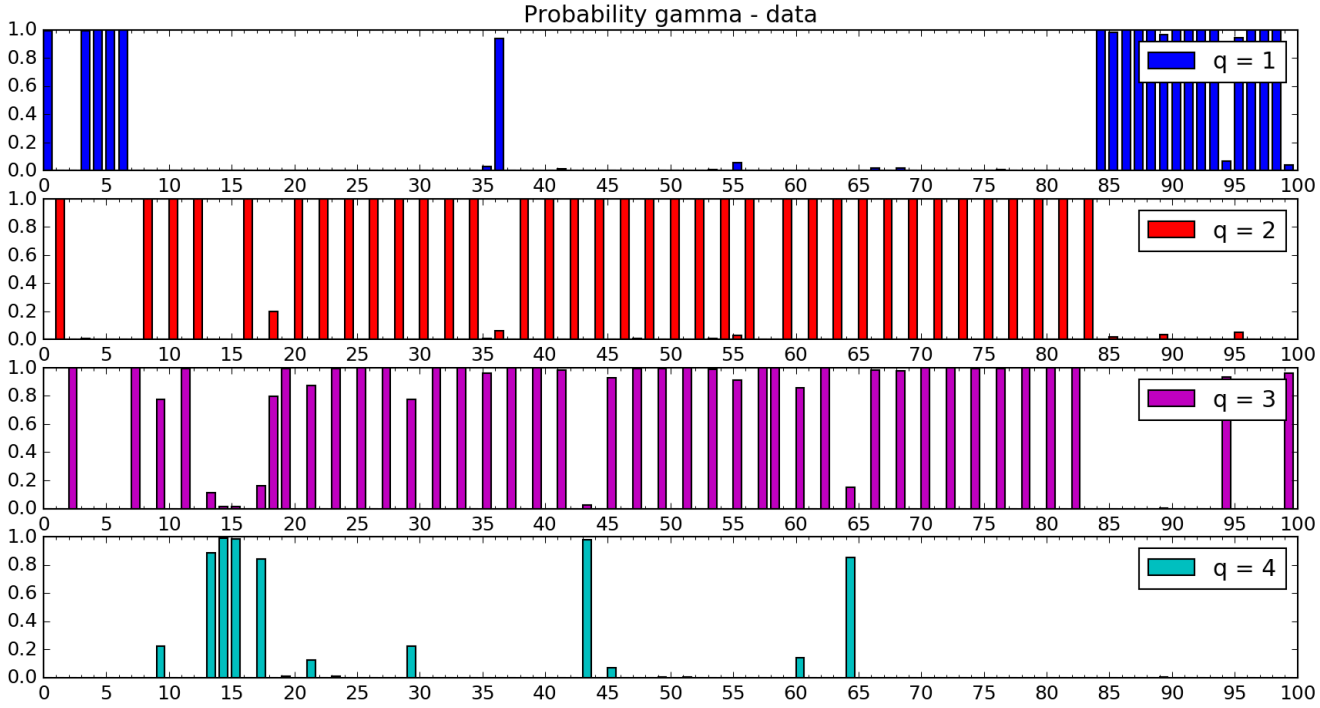

Figure 1: $\gamma_t(q_t) = p(q_t \mid u_1, ..., u_T)$ for the first 100 points of the training dataset.

3 EM algorithm

Now that we have computed the expected probability $\gamma_t(q_t)$ and $\xi_t(q_t, q_{t+1})$ in the previous section we can focus on the estimation of the parameters. We observe that:

$$\begin{aligned}\log p(U, Q \mid \theta) &= \log p(u_1, \dots, u_T, q_1, \dots, q_T \mid \theta) \\ &= \underbrace{\sum_{k=1}^{K} q_1^k \log \pi_k}_{p(q_1)} + \underbrace{\sum_{t=1}^{T-1} \sum_{i=1}^{K} \sum_{j=1}^{K} q_t^i q_{t+1}^j \log a_{ij}}_{p(q_{t+1}|q_t)} + \underbrace{\sum_{t=1}^{T} \sum_{k=1}^{K} q_t^k \log \mathcal{N}(u_t, \mu_k, \Sigma_k)}_{p(u_t|q_t)}\end{aligned}$$

From this we deduce the maximization steps:

$$\begin{aligned}\hat{\pi}_k &= \gamma_1(k) \\ \hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \\ \hat{\mu}_k &= \frac{\sum_{t=1}^{T} u_t \gamma_t(k)}{\sum_{t=1}^{T} \gamma_t(k)} \\ \hat{\Sigma}_k &= \frac{\sum_{t=1}^{T} (u_t - \hat{\mu}_k)(u_t - \hat{\mu}_k)^\intercal \gamma_t(k)}{\sum_{t=1}^{T} \gamma_t(k)}\end{aligned}$$

5 log-likelihood

We plot the log-likelihood as a function of the number of iterations for both dataset and we obtain the figures below:

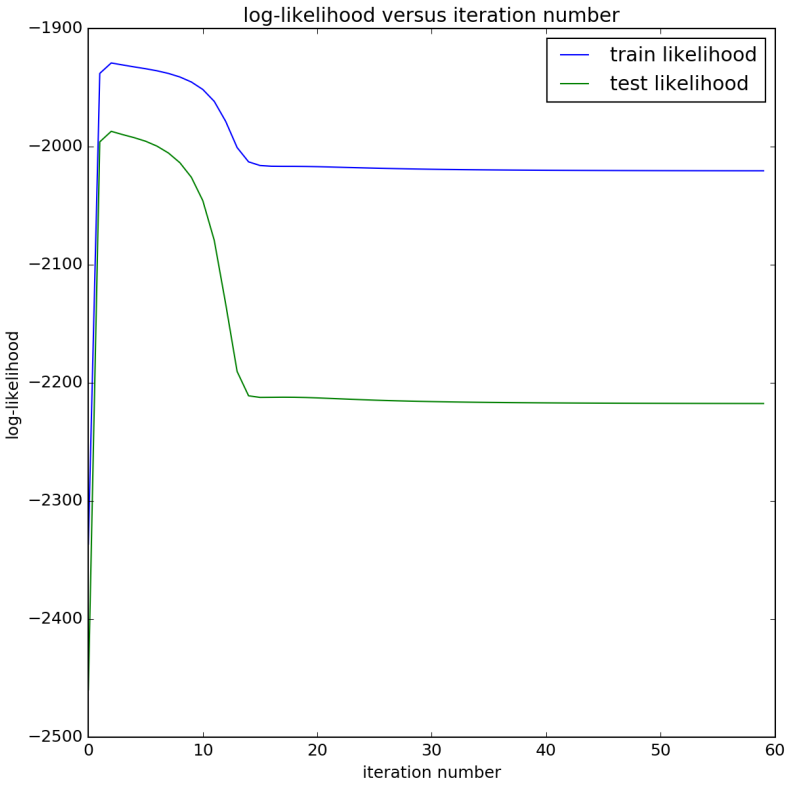

Figure 2: log-likelihood for the train and test dataset.

First we observe that the two have the same overall behavior and that only few iterations are needed for the log-likelihood to converge for both datasets. Finally the train likelihood is better than the test likelihood which isn't a big surprise has the model is specifically trained to fit the train dataset.

6 HMM vs Gaussian Mixture

We have the following

log-likelihood		
Algorithm type	**Training dataset**	**Test dataset**
HMM	-2018	-2215
Gaussian Mixture	-2328	-2409

Although the HMM use a multivariate gaussian for the observed points generation the models are totally different in the sense that the Gaussian Mixture doesn't consider the time transition from one state to the other. This implies that directly comparing the log-likelihood is not really a suited.

7 Viterbi pseudo-code

The Viterbi algorithm stems from the fact that:

$$\begin{aligned}
\max_{q_0,\dots,q_T} p(q_0,\dots,q_T \mid u_0,\dots,u_T) &= \max_{q_0,\dots,q_T} p(q_0,\dots,q_T,u_0,\dots,u_T) \\
&= \max_{q_0,\dots,q_T} \left(p(q_0)p(y_0 \mid q_0) \prod_{t=1}^{T} p(q_t \mid q_{t-1})p(u_t \mid q_t) \right) \\
&= \max_{q_T} \left(\max_{q_0,\dots,q_{T-1}} \left(p(q_0)p(y_0 \mid q_0) \prod_{t=1}^{T} p(q_t \mid q_{t-1})p(u_t \mid q_t) \right) \right) \\
&= \max_{q_T} \left(p(u_T \mid q_T) \max_{q_0,\dots,q_{T-1}} \left(p(q_T \mid q_{T-1})p(q_0)p(y_0 \mid q_0) \prod_{t=1}^{T-1} p(q_t \mid q_{t-1})p(u_t \mid q_t) \right) \right) \\
&= \max_{q_T} \left(p(u_T \mid q_T) \max_{q_{T-1}} \left(p(q_T \mid q_{T-1})p(u_{T-1} \mid q_{T-1}) \dots \left(\max_{q_0} p(q_0)p(y_0 \mid q_0)p(q_1 \mid q_0) \right) \right) \right)
\end{aligned}$$

Now considering only the part $\max_{q_0} p(q_0)p(y_0 \mid q_0)p(q_1 \mid q_0)$, for each value q_1 we compute the best q_0 that maximize this value and we store both the value and the argument.
Then for $\max_{q_1} p(u_1 \mid q_1)p(q_2 \mid q_1) \max_{q_0} p(q_0)p(y_0 \mid q_0)p(q_1 \mid q_0)$ we compute the best q_1 that maximizes it for each q_2 and we iterate which yields the pseudo-code:

for *all* q_1 **do**
  find $q_0^*(q_1)$ that maximizes $\delta_0^*(q_1) = p(q_0)p(u_0 \mid q_0)p(q_1 \mid q_0)$;
  store $\delta_0^*(q_1)$;
  store $q_0^*(q_1)$ in $state^*(q_1)$;
end
for *t = 1 to T-1* **do**
  for *all* q_{t+1} **do**
    find $q_t^*(q_{t+1})$ that maximizes $\delta_t^*(q_{t+1}) = p(u_t \mid q_t)p(q_{t+1} \mid q_t)\delta_{t-1}^*(q_t)$;
    store $\delta_t^*(q_{t+1})$;
    store $q_t^*(q_{t+1})$ by appending it to $state^*(q_{t+1})$;
  end
end
for *all* q_T **do**
  find q_T^* that maximizes $\delta_T^* = p(u_T \mid q_T)\delta_{T-1}^*(q_T^*)$;
  store q_T^* by appending it to $state^*(q_T)$;
end
return $state^*$

Algorithm 1: Viterbi pseudo-code for HMM

8 Viterbi decoding

After implementing the Viterbi decoding we can plot the most likely sequence of states and represent each datapoint of the training dataset and its cluster, in the figure below:

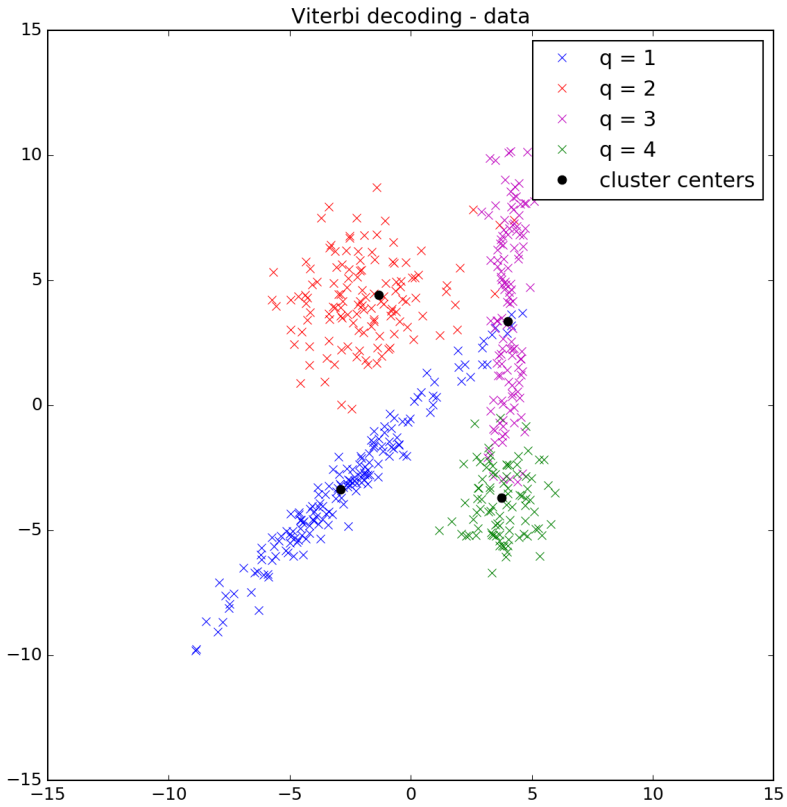

Figure 3: Viterbi decoding for the training dataset.

9 Plot γ for the testing dataset

We can thus plot the probability $\gamma_t(q_t) = p(q_t \mid u_1, ..., u_T)$ for the first 100 points of the testing dataset:

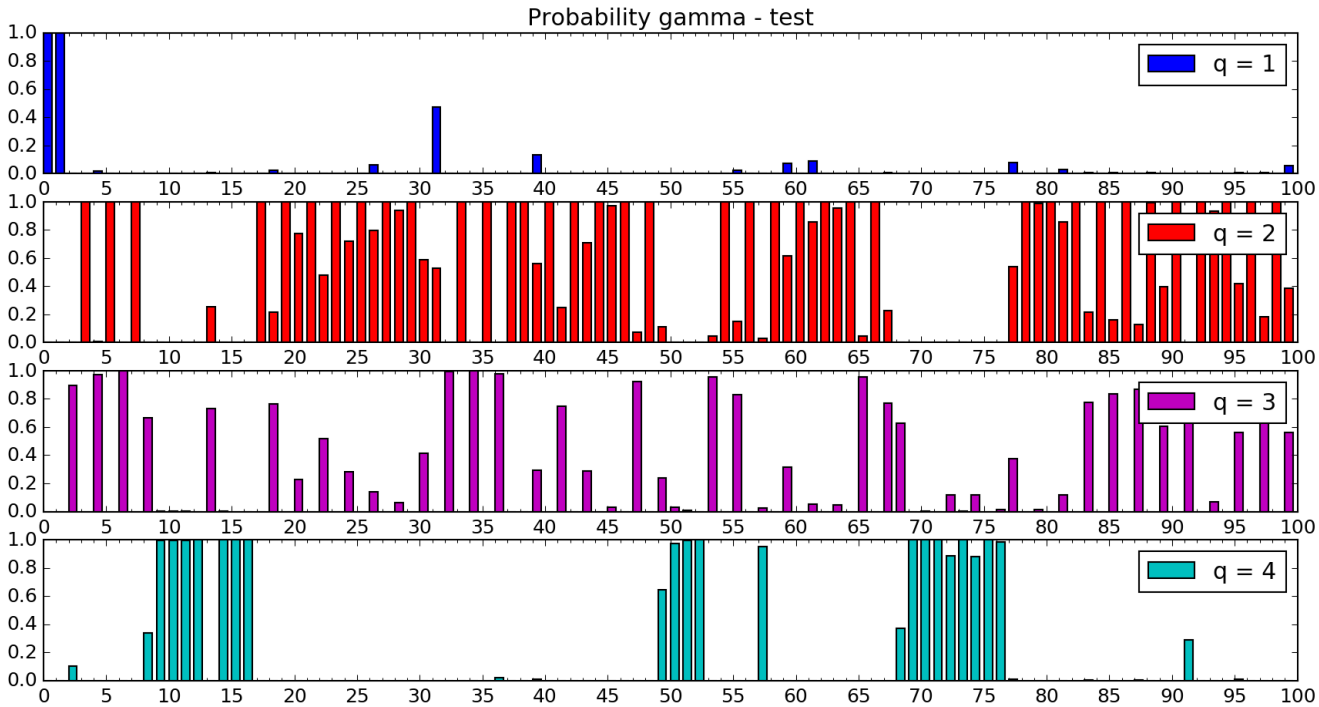

Figure 4: $\gamma_t(q_t) = p(q_t \mid u_1, ..., u_T)$ for the first 100 points of the testing dataset.

10 Marginal probability estimation

Now we plot the hidden variable state for the testing dataset using the marginal probability.

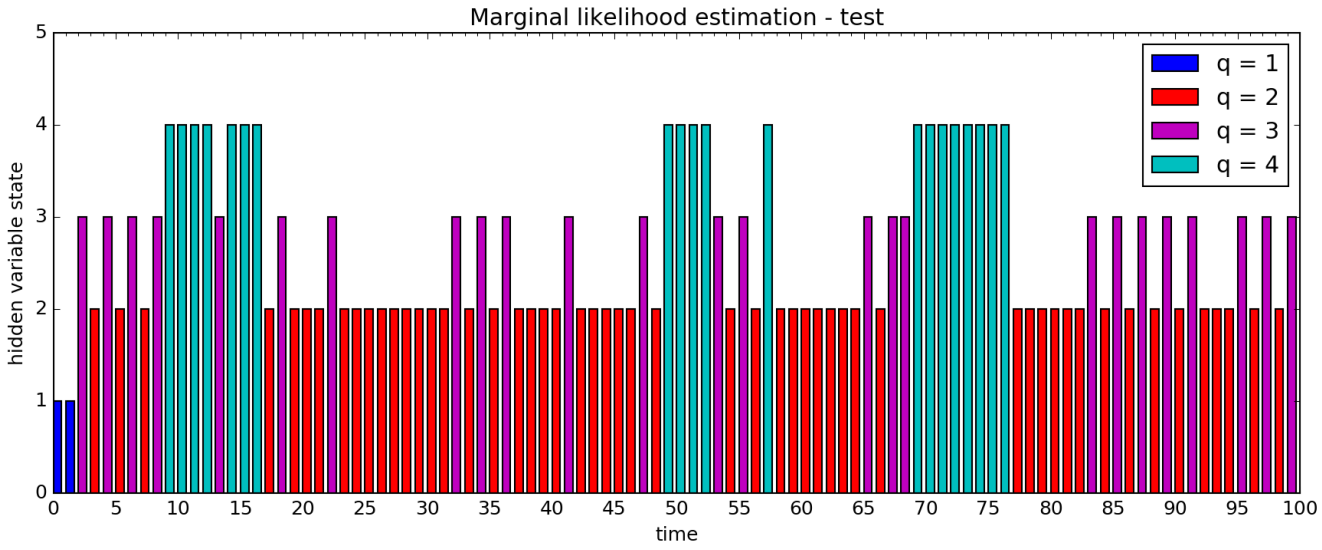

Figure 5: Hidden variable state using the marginal probability.

We can also represent the corresponding point and their cluster:

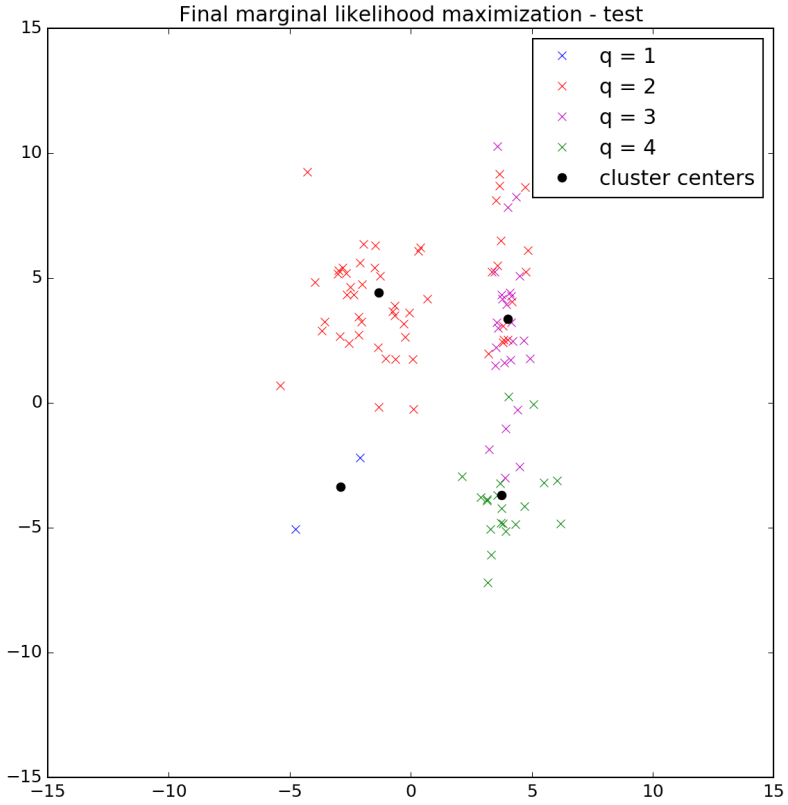

Figure 6: Hidden variable state using the marginal probability.

11 Viterbi decoding

Now we plot the hidden variable state for the testing dataset using the Viterbi decoding.

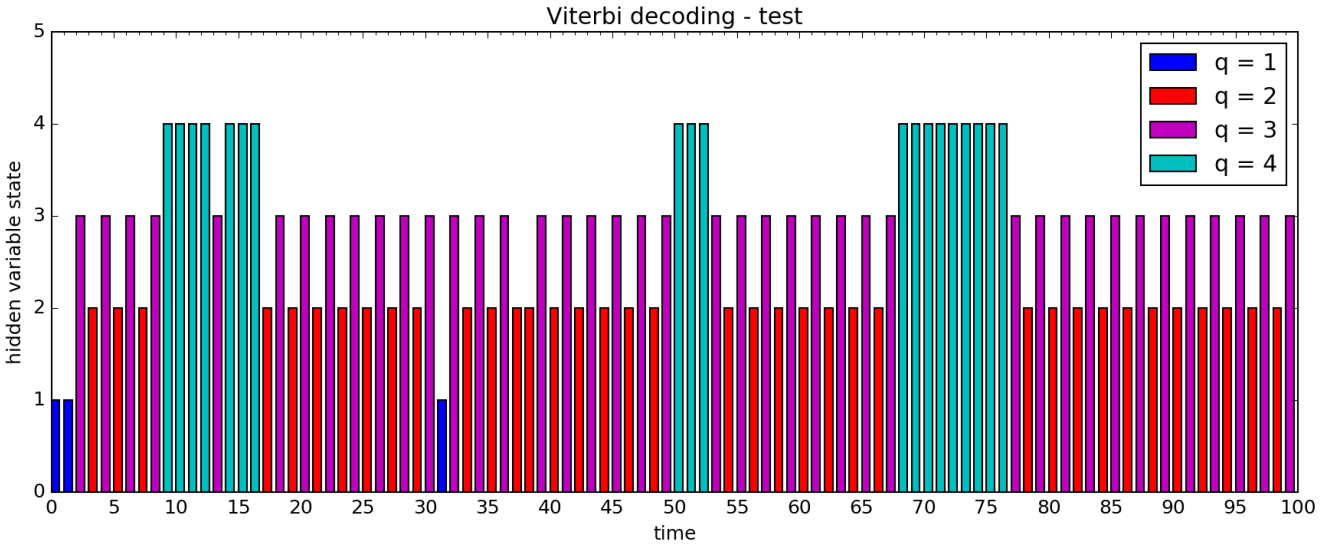

Figure 7: Hidden variable state using the viterbi decoding.

We can also represent the corresponding point and their cluster:

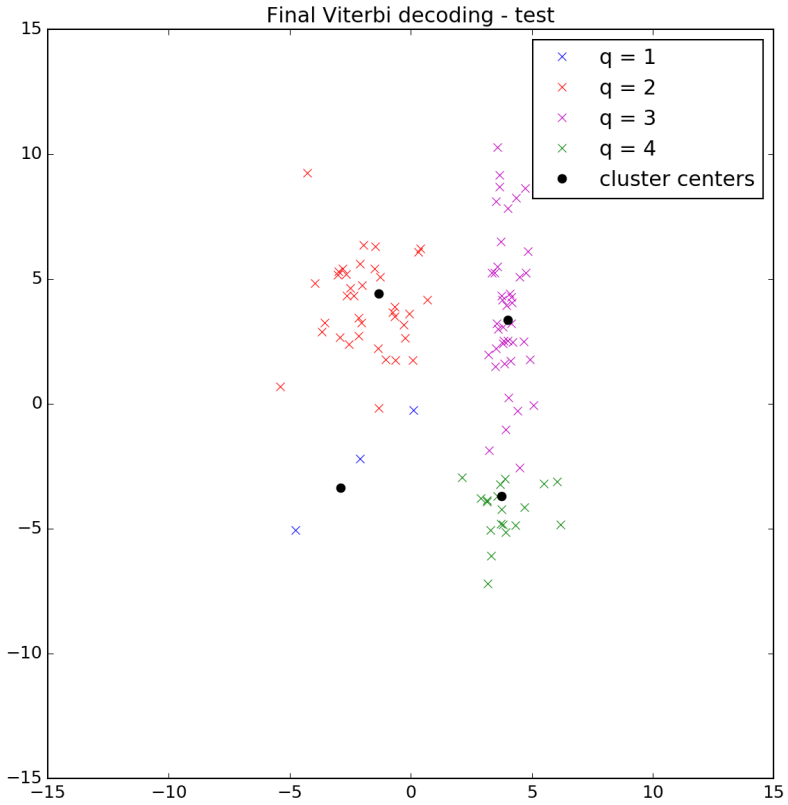

Figure 8: Hidden variable state using the viterbi decoding.

We observe that the two plots are quite different thus the most likely overall sequence is very different from the sequence of most likely states.

12 Number of states

In this exercice the number of states was a given parameter. If we didn't know it we could have used cross-validation to find the most-likely model by trying different state number.