[M2, MVA]

Deep Learning

Maha ELBAYAD

maha.elbayad@student.ecp.fr

Homework 1

November 14, 2015

1 Analytical exercises

1. Considering the 0/1 loss function $L(y, f(x)) = 1 - \mathbb{1}[y = f(x)]$ the Expected prediction error takes the form:

$$EPE(f) = \mathbb{E}_x \left[\sum_y L(y, f(x)\mathbb{P}(y|x)) \right]$$

It suffices to minimize the EPE pointwise, i.e consider:

$$\hat{f}(x) = \arg\min_g \sum_y L(y, g)\mathbb{P}(y|x)$$

Thus

$$\begin{aligned}
\hat{f}(x) &= \arg\min_g \sum_y \left(1 - \mathbb{1}[y = g]\right)\mathbb{P}(y|x) \\
&= \arg\min_g \sum_{y \neq g} \mathbb{P}(y|x) \\
&= \arg\min_g 1 - \mathbb{P}(g|x) \\
&= \arg\max_g \mathbb{P}(g|x)
\end{aligned}$$

2. Let us consider a pair of discrete r.v X and Y on sets $\mathcal{X}$ and $\mathcal{Y}$ respectively, such that $X \perp\!\!\!\perp Y$.

$$\begin{aligned} H(X,Y) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} \mathbb{P}(x,y)\log\mathbb{P}(x,y) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} \mathbb{P}(x)\mathbb{P}(y)(\log\mathbb{P}(x)+\log\mathbb{P}(y)) \\ &= -\sum_{x\in\mathcal{X}} \mathbb{P}(x)\log\mathbb{P}(x)\sum_{y\in\mathcal{Y}}\mathbb{P}(y) - \sum_{y\in\mathcal{Y}}\mathbb{P}(y)\log\mathbb{P}(y)\sum_{x\in\mathcal{X}}\mathbb{P}(x) \\ &= H(X)+H(Y) \end{aligned}$$

2 Data modelling: PCA and K-means

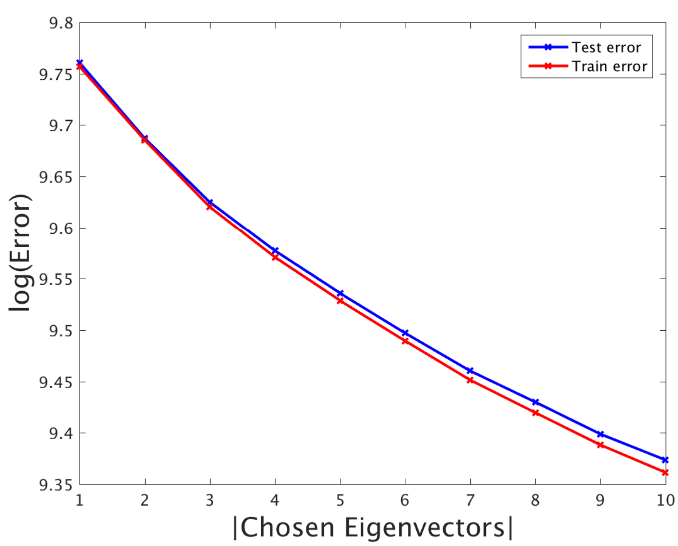
2.1 PCA

We perform the PCA on the training set and estimate the reconstruction error on the test set as:

$$E^{PCA}(D) = \sum_{n\in Testset} \|I_n - \left(\sum_{d=1}^{D} c_{n,k}e_k + m\right)\|_2$$

While varying D: the number of chosen eigenvectors with the largest magnitudes. The results are shown below (figure 1)

Figure 1: Reconstruction error on test set vs train set



2.2 K-means

We run k-means algorithms on the “7” training set. The distortion keeps decreasing after each k-mean iteration (figure 2). To avoid local minimas we run k-means different times with different initializations and keep the optimal clustering.

For $k = 2$ we find the following centroids (figure 3) with some elements of each cluster shown in (figure 4)

Figure 2: K-means Distortion minimization

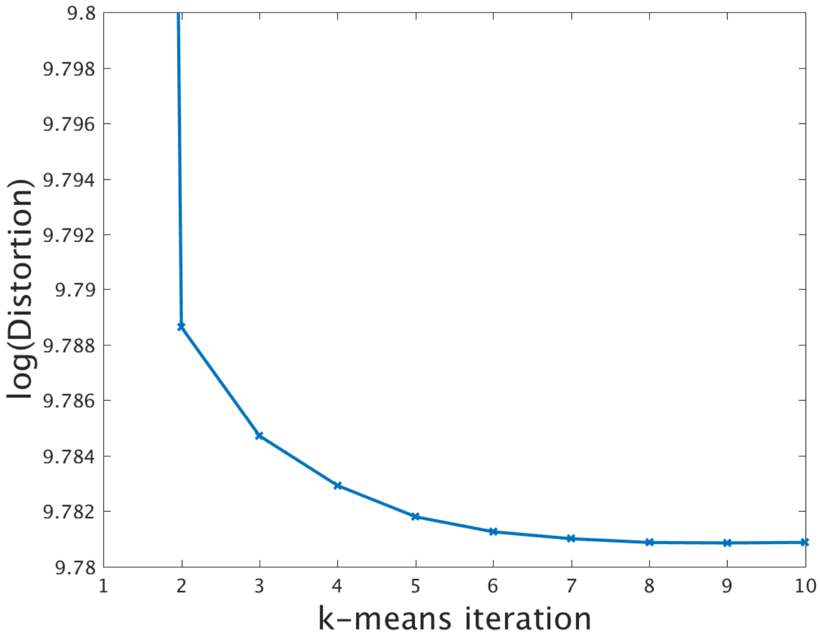
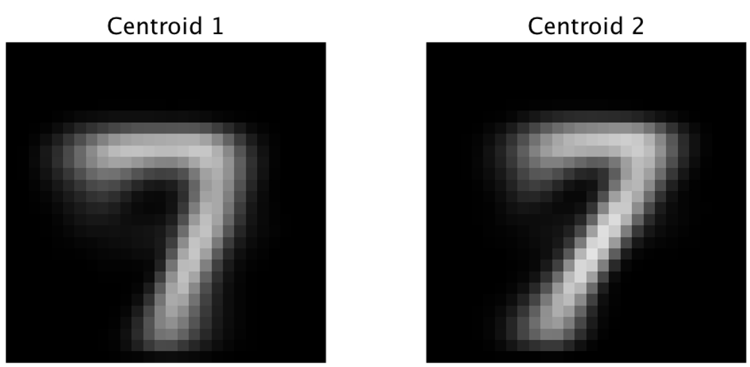
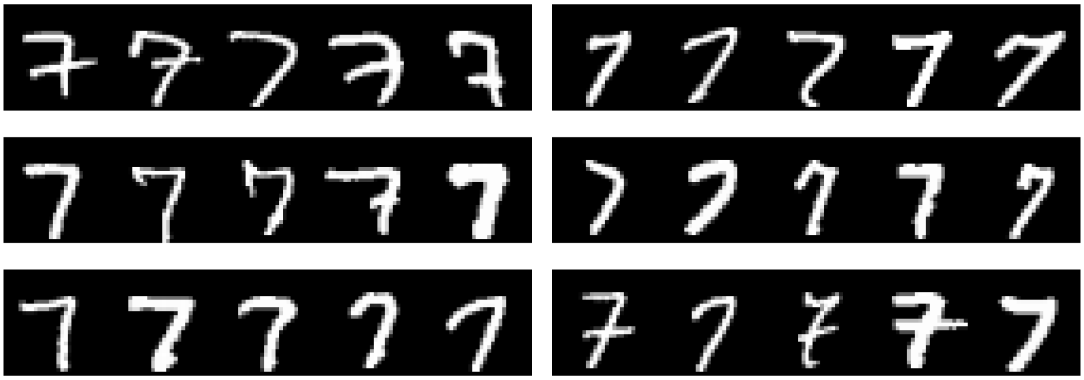


Figure 3: K-means centroids k=2





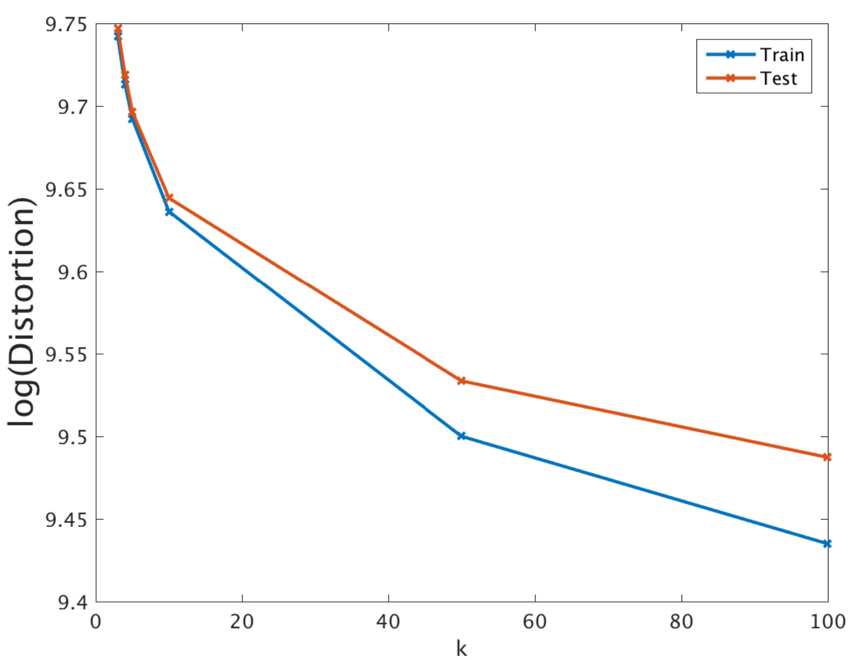
(a) C1 (b) C2

Figure 4: Samples from each cluster - k=2

We repeat the same process for $k \in \{3, 4, 5, 10, 50, 100\}$, the optimal distortion of each model achieved on the train set is shwon in (figure 5) we also evaluate the distortion on the test set for comparison with PCA.

$$E^{KM}(k) = \sum_{n \in Testset} \|I_n - c^{\text{affect}(n)}\|_2$$

Figure 5: K-means distortion on train set and test set



We note that PCA & k-means perform quite similarly on the train set, but k-means fails to generalize to unseen data while PCA does well. However, k-means is sparser (ntest images ⇒ ntest parameters) compared to PCA which is more of a dimensionality reduction technique (this explain the lower error on the test set) and with D chosen eigenvectors we'll have $ntest \times D$ parameters.

3 Ising model: MCMC & learning

3.1 Part-1: Brute-force evaluations

We consider an Ising model on $N \times N$ lattice with N=4. We generate all possible states (2^{N^2})and evaluate the energy at each state x following the formula:

$$E(x, J) = \sum_{(i,j)\in\mathcal{E}} x_i x_j = -\frac{J}{2} x^T \mathbf{A} x$$

Where $J \in \mathbb{R}$ and $\mathcal{E}$ the edges of the network with given incidence matrix $\mathbf{A}$. Each state x occurs with a probability:

$$P(x, J) = \frac{1}{Z} \exp(-E(x, J))$$

with the partition function $Z = \sum_x \exp(-E(x, J))$.
(Figure 7) shows the states with the lowest and largest energy and the probability of each state is given in (figure 6).

Figure 6: States Probabilities

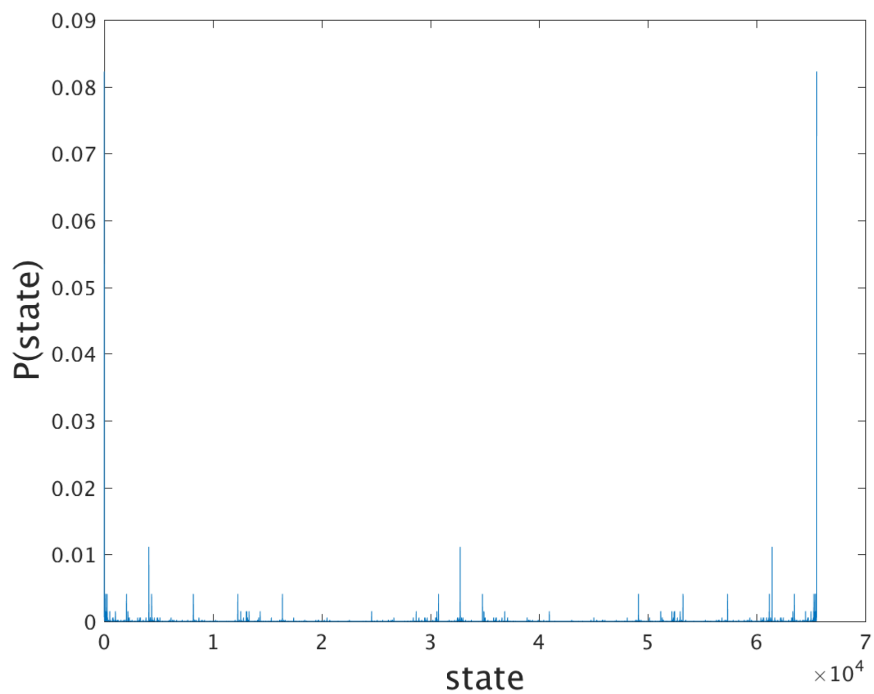
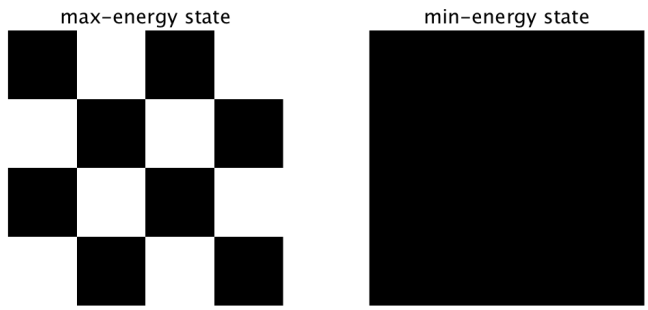


Figure 7: Minimum and maximum energy states

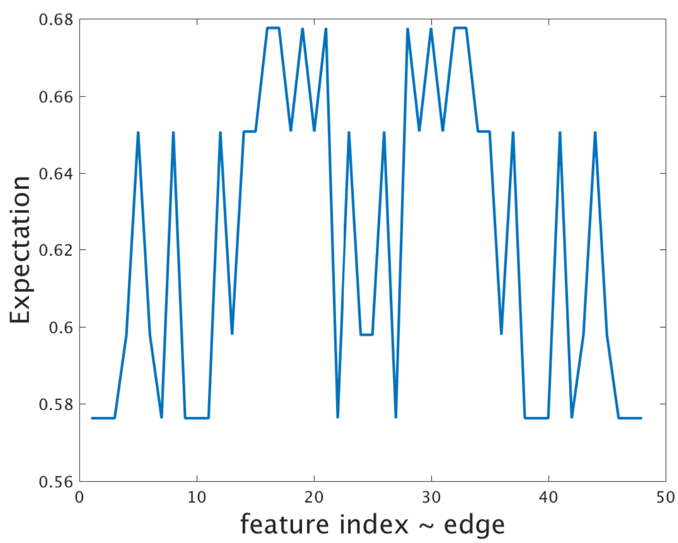


We compute the expectation of each feature $\phi_k(x),\ k = 1, .., K$ under the studied model:

$$E_x^{P_J} = \sum_x \mathbb{P}(x, J)\phi_k(x)$$

Knowing that $\phi_k(x) = x_i x_j$ for $e_k = (i, j) \in \mathcal{E} = \{e_1, ...e_K\}$. The results are shown in (figure 8)

Figure 8: Features expectation - Brute force

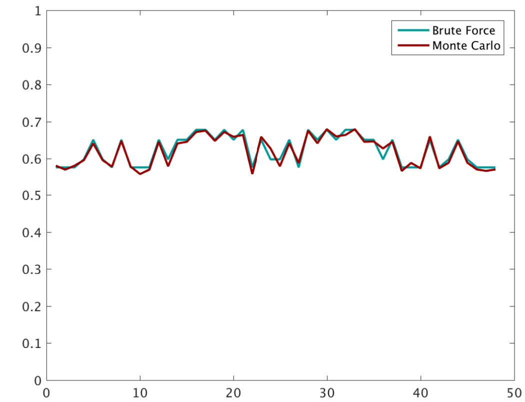


3.2 Part-2: MCMC evaluations

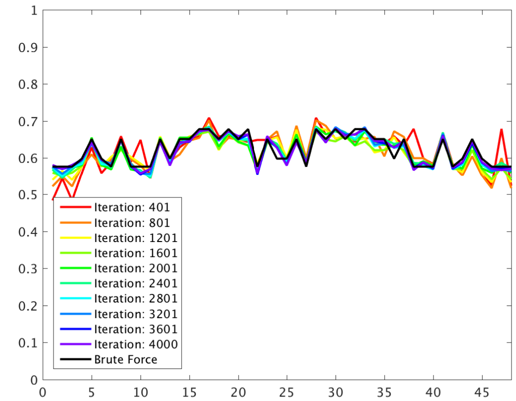
We use Gibbs sampling to obtain a sample of $n = 4000$ states then estimate the expectation of each feature as:

$$E_k^{MC}(n) = \frac{1}{n}\sum_{i=1}^{n} \phi_k(x_i)$$

(Figure 9b) shows the evolution of the expectation while growing the sample and a final comparison to the brute force expectations of the previous section is shown in (figure 9a).



(a) Features expectation - Brute force vs MC



(b) Features expectation - MC's growing sample

3.3 Part-3: Parameter Estimation

In this part we would like to estimate the value of J most likely responsible for given values of E^{P_J}. Hence we will maximize $\log \mathbb{P}_J(X)$ where $X = \{x_1, ..x_N\}$ is the training sample.
To do so we use gradient ascent:

$$J := J - \delta \frac{\partial \log \mathbb{P}_J(X)}{\partial J}$$

Where δ is a learning rate.
Given that:

$$\log \mathbb{P}_J(X) = -\log Z + \sum_n^N w\phi(x_n), \;\; Z = \sum_x \exp(w\phi(x))$$

Where $w = -J$ and $\phi(x) = \sum_{e=(i,j)\in\mathcal{E}} x_i x_j$

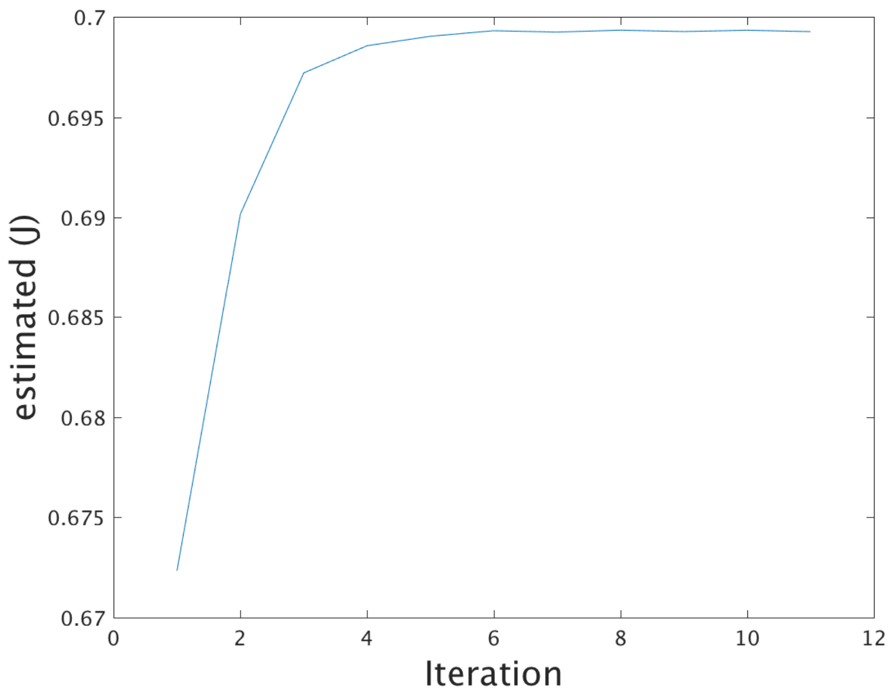
$$\begin{aligned}\frac{\partial \log \mathbb{P}_J(X)}{\partial J} &= \sum_{n=1}^N \phi(x_n) - \frac{N}{Z}\sum_x \phi(x)\exp(w\phi(x)) \\ &= N(<\phi>_{emp} - <\phi>_{\mathbb{P}_J})\end{aligned}$$

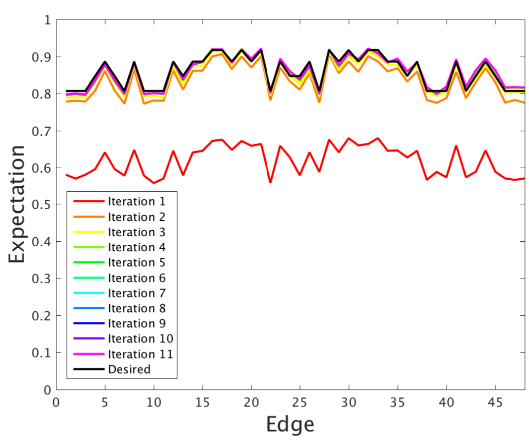
Hence the update:

$$J := J - \delta(<\phi>_{emp} - <\phi>_{\mathbb{P}_J})$$

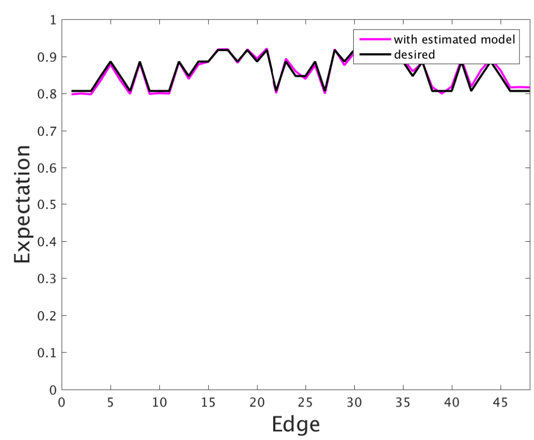
Results: starting from J=.5 we reach maximum likelihood at $J = .6993$, The J updates are shown in (figure 9). (Figure 10a) shows the features expectations $< x_i x_j >_{emp}$ after each gradient ascent step until optimal solution is found (figure 10b).

Figure 9: J gradient ascent





(a) Expectations at each iteration



(b) Estimated expectations vs desired expectations