Introduction to Machine Learning (67577)

Exercise 1
Estimation Theory & Mathematical Background

Second Semester, 2022

Contents

1 Submission Instructions

Please make sure to follow the general submission instructions available on the course website. In addition, for the following assignment, submit a single `ex1_ID.tar` file containing:

- An `Answers.pdf` file with the answers for all theoretical and practical questions (include plotted graphs *in* the PDF file).
- The following python files (without any directories): `gaussian_estimators.py`, `fit_gaussian_estimators.py`

The `ex1_ID.tar` file must be submitted in the designated Moodle activity prior to the date specified *in the activity*.

- Late submissions will not be accepted and result in a zero mark.
- Plots included as separate files will be considered as not provided.
- Do not forget to answer the Moodle quiz of this assignment.

2 Theoretical Part

2.1 Mathematical Background

2.1.1 Linear Algebra

1. Prove that orthogonal matrices are isometric transformations. That is, let $T : V \mapsto W$ be some linear transformation and A the corresponding matrix. Show that if A is an orthogonal matrix then $\forall x \in V \;\; ||Ax|| = ||x||$.

   > $$||Ax||^2 = (Ax)^\top (Ax) = x^\top A^\top Ax = x^\top Ix = ||x||^2$$

2. Calculate the SVD of the following matrix A. That is, find the matrices $U, \Sigma, V^\top$ where U, V are orthogonal matrices and Σ diagonal.

   $$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & -1 & 2 \end{bmatrix}$$

   Recall, that to find the SVD of A we can calculate $A^\top A$ to deduce V, Σ and then calculate $AA^\top$ to deduce U. Equivalently, once we deduced V, Σ we can fine U using the equality $AV = U\Sigma$.

   > First we find that
   >
   > $$A^\top A = \begin{bmatrix} 2 & 0 & 2 \\ 0 & 2 & -2 \\ 2 & -2 & 4 \end{bmatrix}$$
   >
   > and search for its eigenvalues by solving $det(A^\top A - \lambda I) = 0$. This yields the equation $-\lambda^3 + 8\lambda^2 - 12\lambda = 0$ from which we derive the eigenvalues $\lambda = 6, 2, 0$. The (normalized) eigenvectors corresponding these eigenvalues are:
   >
   > $$\mathbf{v}_1 = \begin{bmatrix} 1/\sqrt{6} \\ -1/\sqrt{6} \\ 2/\sqrt{6} \end{bmatrix}, \quad \mathbf{v}_2 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{bmatrix}, \mathbf{v}_3 = \begin{bmatrix} -1/\sqrt{3} \\ 1/\sqrt{3} \\ 1/\sqrt{3} \end{bmatrix}$$

corresponding $\lambda_1 = 6, \lambda_2 = 2, \lambda_3 = 0$. Therefore we derive that:

$$\Sigma = \begin{bmatrix} \sqrt{6} & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix}, \quad V = \begin{bmatrix} 1/\sqrt{6} & 1/\sqrt{2} & -1/\sqrt{3} \\ -1/\sqrt{6} & 1/\sqrt{2} & 1/\sqrt{3} \\ 2/\sqrt{6} & 0 & 1/\sqrt{3} \end{bmatrix}$$

To find the matrix U we calculate $AA^\top$ and find the eigenvectors corresponding the eigenvalues. Another way is to notice for $A = U\Sigma V^\top$, since V is an orthogonal matrix, then by multiplying from the right by V we get that $AV = U\Sigma$. Solving this system we get that:

$$U = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} \sqrt{6} & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix}, \quad V = \begin{bmatrix} 1/\sqrt{6} & 1/\sqrt{2} & -1/\sqrt{3} \\ -1/\sqrt{6} & 1/\sqrt{2} & 1/\sqrt{3} \\ 2/\sqrt{6} & 0 & 1/\sqrt{3} \end{bmatrix}$$

3. In this question we prove the Power-Iteration algorithm for finding the SVD of a matrix. Let $A \in \mathbb{R}^{m\times n}$ and define $C_0 = A^\top A$. Denote $\lambda_1 \geq \ldots \geq \lambda_n$ the eigenvalues of C_0, with the corresponding normalized eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$.

   Let us assume the $\lambda_1 > \lambda_2$. Define $b_k \in \mathbb{R}$ as follows:

$$b_0 = \sum_{i=1}^{n} a_i v_i, \quad b_{k+1} = \frac{C_0 b_k}{\|C_0 b_k\|}$$

   where $a_1 \neq 0$. Show that: $\lim_{k\to\infty} b_k = \pm v_1$.

Define $b_0 := \sum c_i v_i$. Then:

$$\begin{aligned} A^k b_0 &= \sum c_i \cdot A^k v_i \\ &= \sum c_i A^{k-1} (Av_i) \\ &= \sum c_i A^{k-1} (\lambda_i v_i) \\ &= \sum c_i \lambda_i^k v_i \\ &= c_1 \lambda_1^k \left(v_1 + \sum_{i=2} \tfrac{c_i}{c_1} \left(\tfrac{\lambda_i}{\lambda_1} \right)^k v_i \right) \end{aligned}$$

Therefore, as we increase k:

$$\begin{aligned} \lim_{k\to\infty} A^k b_0 &= \lim_{k\to\infty} c_1 \lambda_1^k v_1 + c_1 \lambda_1^k \sum_{i=2} \lim_{k\to\infty} \left(\tfrac{\lambda_i}{\lambda_1} \right)^k c_i v_i \\ &= c_1 \lambda^k v_1 + c_1 \lambda_1^k \sum_{i=2} 0 \cdot c_i v_i \\ &= c_1 \lambda^k v_1 \end{aligned}$$

As for any $i > 1$ it holds that $(\lambda_i/\lambda_1) < 1$.

2.1.2 Multivariate Calculus

4. Let $x \in \mathbb{R}^n$ be a fixed vector and $U \in \mathbb{R}^{n\times n}$ a fixed orthogonal matrix. Calculate the Jacobian of the function $f : \mathbb{R}^n \to \mathbb{R}^n$:

$$f(\sigma) = U \cdot \operatorname{diag}(\sigma) U^\top x$$

Where $\operatorname{diag}(\sigma)$ is an $n \times n$ matrix where

$$\operatorname{diag}(\sigma)_{ij} = \begin{cases} \sigma_i & i = j \\ 0 & i \neq j \end{cases}$$

> We can express $f(\boldsymbol{\sigma})$ as
>
> $$f(\boldsymbol{\sigma}) = \sum_{i=1}^{n} \sigma_i \boldsymbol{u}_i \boldsymbol{u}_i^\top \boldsymbol{x},$$
>
> where $\boldsymbol{u}_i$ is the i'th column of U. From here it's easy to see that $\frac{\partial f_j(\boldsymbol{\sigma}_0)}{\partial \sigma_i} = [u_i u_i^\top \boldsymbol{x}]_j$, which means the i 'th column of $J_{\boldsymbol{\sigma}}(f)$ is $u_i \cdot \langle u_i, \boldsymbol{x} \rangle$. In matrix notation we can write
>
> $$J_{\boldsymbol{\sigma}}(f) = U \cdot \operatorname{diag}\left(U^\top \boldsymbol{x}\right)$$

5. Use the chain rule to calculate the gradient of $h(\sigma) = \frac{1}{2} ||f(\sigma) - y||^2$

> Begin with expressing h as follows:
>
> $$h(\sigma) = \frac{1}{2} f ||(\sigma)||^2 - f(\sigma)^\top y + \frac{1}{2} ||y||^2$$
>
> Now, using the chain rule then
>
> $$\begin{aligned} \frac{\partial h}{\partial \sigma_i} = \frac{\partial h}{\partial f} \frac{\partial f}{\partial \sigma_i} &= \frac{\partial}{\partial f} \left(\frac{1}{2} ||f(\sigma)||^2 - f(\sigma)^\top y + \frac{1}{2} ||y||^2 \right) \cdot \frac{\partial f}{\partial \sigma_i} \\ &= (f(\sigma) - y)^\top \cdot \frac{\partial f}{\partial \sigma_i} \end{aligned}$$
>
> Notice that $\frac{\partial f}{\partial \sigma_i}$ is in fact the Jacobian of f which we have calculated in the previous question. Therefore:
>
> $$\nabla h = (f(\sigma) - y)^\top J_\sigma(f)$$

6. Calculate the Jacobian of the softmax function $S : \mathbb{R}^d \to [0,1]^k$

$$S(\mathbf{x})_j = \frac{e^{x_j}}{\sum_{l=1}^{k} e^{x_l}}$$

> Recall that for a multivariate function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^m$ the Jacobian is the $k \times d$ matrix of all partial derivatives such that
>
> $$[J_{\mathbf{x}}(\mathbf{f})]_{ij} = [\nabla \mathbf{f}(\mathbf{x})_i]_j = \frac{\partial f_i(\mathbf{x})}{\partial x_j}$$
>
> Therefore, in the case of the Softmax function then:
>
> $$[\nabla S(\mathbf{x})_i]_j = \frac{\partial}{\partial x_j} \frac{e^{x_i}}{\sum_l^k e^{x_l}} = \frac{e^{x_i} (\sum_l^k e^{x_l}) - e^{x_i} e^{x_j}}{(\sum_l^k e^{x_l})^2}$$

where, if $i = j$ then:

$$[\nabla S(\mathbf{x})_i]_j = \frac{e^{x_i}}{\sum_l^k e^{x_l}} \cdot \frac{(\sum_l^k e^{x_l}) - e^{x_j}}{\sum_l^k e^{x_l}} = S(\mathbf{x})_i\,(1 - S(\mathbf{x})_i)$$

and if $i \neq j$ then:

$$[\nabla S(\mathbf{x})_i]_j = \frac{0 \cdot \sum_l^k e^{x_l} - e^{x_i} e^{x_j}}{(\sum_l^k e^{x_l})^2} = -\frac{e^{x_i}}{\sum_l^k e^{x_l}} \cdot \frac{e^{x_j}}{\sum_l^k e^{x_l}} = -S(\mathbf{x})_i S(\mathbf{x})_j$$

this can be written as

$$[\nabla S(\mathbf{x})_i]_j = S(\mathbf{x})_i(\delta_{ij} - S(\mathbf{x})_j), \quad \delta_{ij} = \mathbb{1}_{[i=j]}$$

which in matrix notation yields:

$$J_{\mathbf{x}}(S) = \mathrm{diag}(S) - SS^\top$$

7. Let $f : \mathbb{R}^d \to \mathbb{R}$ be defined as $f(x,y) = x^3 - 5xy - y^5$. Calculate the Hessian of f.

We begin with calculating the gradient of f:

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)^\top = \left(3x^2 - 5y, -5y^4 - 5x\right)^\top$$

Next, for each of the partial derivatives we calculate a second derivative with respect to each of the parameters. So:

$$H_f = \begin{bmatrix} 6x & -5 \\ -5 & -20y^3 \end{bmatrix}$$

2.2 Estimation Theory

8. Let $x_1, x_2, \ldots \overset{iid}{\sim} \mathcal{P}$ be a sample of infinity size drawn from some probability distribution function $\mathcal{P}$ with finite expectation and variance. Show that the sample mean estimator $\hat{\mu}_n = \frac{1}{n}\sum x_i$ calculated over the first n samples is a consistent estimator. Hint: for any given fixed value of $n \in \mathbb{N}$ bound from above the probability of deviating more than ε.

To prove that $\hat{\mu}_n$ is a consistent estimator, we must show that

$$\forall \varepsilon > 0 \quad \lim_{n\to\infty} \mathbb{P}\left(|\hat{\mu}_n - \mu| > \varepsilon\right) = 0$$

Notice, that as the sample mean is an unbiased estimator then $\forall n \in \mathbb{N}$ it holds that $\mathbb{E}[\hat{\mu}_n] = \mu$. Thus, we can use Chebyshev's inequality to bound from above the probability of deviating more than ε for any $\varepsilon > 0$.

The variance of the sample mean estimator is:

$$Var(\hat{\mu}_n) = Var\left(\tfrac{1}{n}\sum_i x_i\right) \overset{iid}{=} \frac{1}{n^2}\sum_i Var(x_i) = \frac{\sigma^2}{n}$$

Therefore, using the sample mean's variance and Chebyshev's inequality we see that:

$$\begin{aligned}\lim_{n\to\infty}\mathbb{P}\left(|\hat{\mu}_n - \mu| \geq \varepsilon\right) &\leq \lim_{n\to\infty}\frac{Var(\hat{\mu}_n)}{\varepsilon}\\ &= \lim_{n\to\infty}\frac{1}{\varepsilon}\cdot\frac{\sigma^2}{n}\\ &= \frac{\sigma^2}{\varepsilon}\cdot\lim_{n\to\infty}\frac{1}{n}\to 0\end{aligned}$$

and as such we conclude that the sample mean is a consistent estimator.

9. Let $\mathbf{x}_1,\dots,\mathbf{x}_m \overset{iid}{\sim} \mathcal{N}(\mu,\Sigma)$ be m observations sampled i.i.d from a multivariate Gaussian with expectation of $\mu\in\mathbb{R}^d$ and a covariance matrix $\Sigma\in\mathbb{R}^{d\times d}$. Provide an expression for the log-likelihood function of $\mathcal{N}(\mu,\Sigma)$. Develop the expression as much as you can. Hint: follow the approach used to derive the likelihood function for the univariate case.

$$\begin{aligned}\mathcal{L}\left(\mu,\Sigma|\mathbf{x}_1,\dots,\mathbf{x}_m\right) &= \mathcal{N}\left(\mathbf{x}_1,\dots,x_m;\mu,\Sigma\right)\\ &\overset{i.i.d}{=} \prod\mathcal{N}\left(\mathbf{x}_i;\mu,\Sigma\right)\\ &= \prod\frac{1}{\sqrt{(2\pi)^d|\Sigma|}}\exp\left(-\tfrac{1}{2}\left(\mathbf{x}_i-\mu\right)^\top\Sigma^{-1}\left(\mathbf{x}_i-\mu\right)\right)\\ &= \left((2\pi)^d|\Sigma|\right)^{-m/2}\cdot\exp\left(-\tfrac{1}{2}\textstyle\sum_i\left(\mathbf{x}_i-\mu\right)^\top\Sigma^{-1}\left(\mathbf{x}_i-\mu\right)\right)\end{aligned}$$

Therefore, the log-likelihood is:

$$\ell\left(\mu,\Sigma|\mathbf{x}_1,\dots,\mathbf{x}_m\right) = -\frac{m}{2}\log\left((2\pi)^d\right) - \frac{m}{2}\log|\Sigma| - \frac{1}{2}\sum_i\left(\mathbf{x}_i-\mu\right)^\top\Sigma^{-1}\left(\mathbf{x}_i-\mu\right)$$

3 Practical Part

Before starting the practical part please make sure to have cloned the IML.HUJI GitHub repository and setup the virtual environment as specified in the instructions. Write the necessary code in the files specified in the questions.

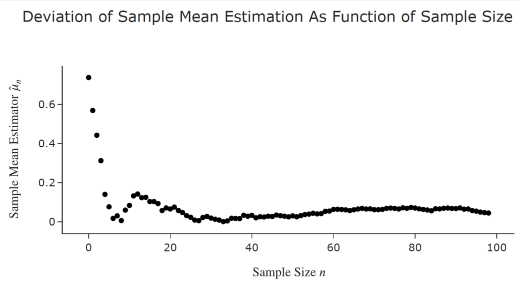
3.1 Univariate Gaussian Estimation

Implement the `UnivariateGaussian` class in the `learners.gaussian_estimators.py` file. Follow details specified in class and function documentation.

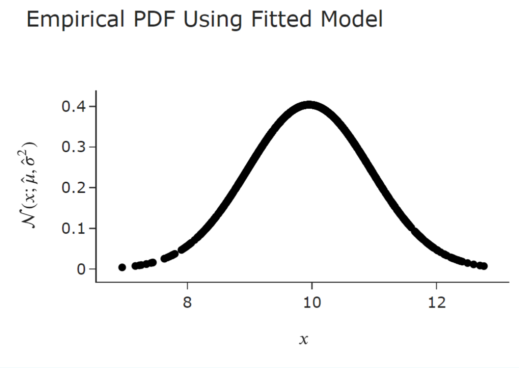
1. Using `numpy.random.normal` draw 1000 samples $x_1,\dots,x_{1000} \overset{iid}{\sim} \mathcal{N}(10,1)$ and fit a univariate Gaussian. Print the estimated expectation and variance. Output format should be `(expectation, variance)`.

Rounded up to 3 decimal places: `(9.955, 0.975)`

2. Over previously drawn samples, fit a series of models of increasing samples size: 10, 20,...,100, 110,...1000. Plot the absolute distance between the estimated- and true value of the expectation, as a function of the sample size. Provide meaningful axis names and title.



3. Compute the PDF of the previously drawn samples using the model fitted in question 1. Plot the empirical PDF function under the fitted model. That is, create a scatter plot with the ordered sample values along the x-axis and their PDFs (using the `UnivariateGaussian.pdf` function) along the y-axis. Provide meaningful axis names and title. What are you expecting to see in the plot?



3.2 Multivariate Gaussian Estimation

Implement the `Multivariate` class in the `learners.gaussian_estimators.py` file. Follow details specified in class and function documentation. When implementing the `log_likelihood` function use the expressions developed in the theoretical part.

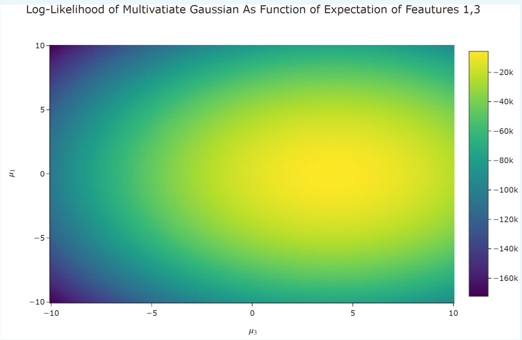
4. Using `numpy.random.multivariate_normal` draw 1000 samples $\mathbf{x}_1,\dots,\mathbf{x}_{1000} \overset{iid}{\sim} \mathcal{N}(\mu,\Sigma)$

$$\mu = \begin{bmatrix} 0 \\ 0 \\ 4 \\ 0 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 1 & 0.2 & 0 & 0.5 \\ 0.2 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0.5 & 0 & 0 & 1 \end{bmatrix}$$

Fit a multivariate Gaussian and print the estimated expectation and covariance matrix. Print each in a separate line.

> Rounded to 3 decimal places:
>
> $$\hat{\mu} = \begin{bmatrix} -0.023 \\ -0.043 \\ 3.993 \\ -0.02 \end{bmatrix}, \quad \hat{\Sigma} = \begin{bmatrix} 0.917 & 0.166 & -0.03 & 0.463 \\ 0.166 & 1.974 & -0.006 & 0.046 \\ -0.03 & -0.006 & 0.98 & -0.02 \\ 0.463 & 0.046 & -0.02 & 0.973 \end{bmatrix}$$

5. Using the samples drawn in the question above calculate the log-likelihood for models with expectation $\mu = [f_1, 0, f_3, 0]^\top$ and the true covariance matrix defined above, where f_1, f_3 get values returned from `np.linspace(-10, 10, 200)`. Plot a heatmap of $f1$ values as rows, f_3 values as columns and the color being the calculated log likelihood. Provide meaningful axis names and title. What are you able to learn from the plot?



6. Of all values tested in question 5, which model (pair of values for feature 1 and 3) achieved the maximum log-likelihood value? Round to 3 decimal places

> Setup of maximum likelihood (features 1 and 3): `[-0.05 3.97]`