Machine Learning Primer — 21.10.2024
WiSe 24/25 — **C. Gros**, D. Nevermann

Exercise Sheet #2

Problem 1 (*PIP and Virtual Environments*)

One of the most powerful things about Python is that there is a library (or package) for almost everything. During the semester we will work with various libraries, however, they first must be installed.
The most common and easiest way to install libraries is using the *PIP package manager*. Usually PIP is installed alongside Python. To test if you already have PIP installed, run `$ pip --version` from the command line.[1] To install a package, you can simply run `$ pip install <package>`.
To avoid conflicts between libraries installed for different projects, it is a common practice to create a *virtual environment* for every project. At the core, this is just a copy of the Python interpreter and the PIP executable. Newly installed packages are placed directly in the directory of the project and are not visible to other projects.
Create a virtual environment and install a package using PIP. Follow these steps:

- To create a virtual environment, navigate to a new project's directory and run `$ python3 -m venv ./venv` from the command line.
- After creating the environment, you still have to activate it using

  ```
  $ source venv/bin/activate          # MacOS or Linux
  PS C:\> venv\Scripts\Activate.ps1    # PowerShell
  ```

  You should now see a `(venv)` in front of your prompt.
- Run `$ pip --version` to check that everything worked. It should tell you that the instance of PIP that is used is located in your project's directory.
- Now install a package! Run `$ pip install numpy`. This should install the *NumPy* package, a popular framework for numerical calculations in Python.

[1] If problems arise try `pip3` instead or check the name of the executable. Also make sure that you have PIP installed.

Problem 2 (*Jupyter Notebooks*)

A nice feature in Python is the ability to run code in so called *Jupyter notebooks*. Notebooks are comprised of cells that can be run individually and also provide Markdown cells (formatted text or even LaTeX- equations) right next to code cells.
Begin by installing JupyterLab using `$ pip install jupyterlab`. Then, from the command line, run `$ jupyter lab`. This should automatically open a browser window that greets you with a nice user interface.

- The interface consists of a file browser on the left, the main work area in the center, and various tabs and panels for different tools and extensions.

- To create a new Jupyter notebook, click the "+ Notebook" button in the file browser or go to "File" > "New" > "Notebook" from the menu bar.

- The notebook interface allows you to write and execute code in individual cells. You can choose the cell type as "Code" or "Markdown" using the dropdown menu in the toolbar. Code cells are used to write and run code, while Markdown cells are used for text and documentation.

- To run the code in a cell, you can either click the "Run" button in the toolbar or use the keyboard shortcut "Shift + Enter". The output will be displayed directly below the cell.

- You can add new cells by clicking the "+" button in the toolbar or by using the keyboard shortcuts "B" to add a cell below the current cell or "A" to add a cell above the current cell.

- Plot something in a notebook using `matplotlib`, a popular plotting library. You first have to install the package using PIP (see Problem 1). Here is a tutorial to get you started.

Problem 3 (*Distributions and Kullback-Leibler Divergence*)

Consider the distributions (probability density functions, PDFs)

$$p(x) = N \cos^2(x) \mathrm{e}^{-\lambda x^2} \qquad \text{(Gaussian-like distribution)}$$
$$q(x) = \frac{\alpha}{(1+x)^{\alpha+1}} \qquad \text{(Pareto distribution)}$$

with support $x \in \mathbb{R}_+$, where $\lambda, \alpha > 0$ and N is the normalization constant.

(a) Show that the normalization constant evaluates to

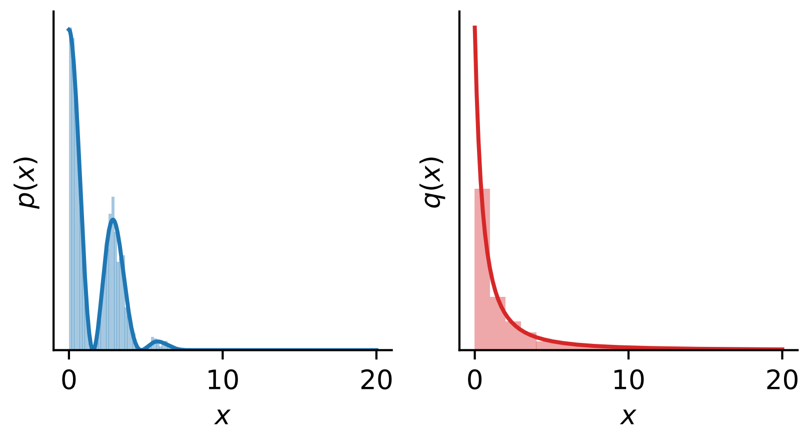
$$N = \frac{4\sqrt{\lambda}}{(1+\exp(-1/\lambda))\sqrt{\pi}}.$$

Hint: If you are aiming for an analytical solution, the identities

$$\int_{-\infty}^{\infty} \mathrm{e}^{-ax^2-bx}\, dx = \sqrt{\frac{\pi}{a}}\mathrm{e}^{\frac{b^2}{4a}} \quad \text{and} \quad \cos(x) = \frac{1}{2}\left(\mathrm{e}^{\mathrm{i}x} + \mathrm{e}^{-\mathrm{i}x}\right)$$

might be helpful.

(b) Implement both distributions in Python and draw $n = 2000$ random numbers from each of the distributions. Plot a histogram of the random numbers you generated together with a plot of the respective PDFs. The result should look something like this:



(c) Compute the Kullback-Leibler divergence $K[p,q]$ numerically as a function of λ. What problem might arise when computing $K[q,p]$?