Homework 1 – Deep Learning (CS/DS541, Murai, Spring 2024)

(Adapted from Prof. Whitehill)

In the problems below, marks can be discounted if the solution is too inefficient (e.g., unnecessary use of for loops) or numerically unstable.

1. **Python and Numpy Warm-up Exercises** This part of the homework is intended to help you review your linear algebra and learn (or refresh your understanding of) how to implement linear algebraic and statistical operations in Python using `numpy` (to which we refer in the code below as `np`). For each of the problems below, write a method (e.g., `problem_1a`) that returns the answer for the corresponding problem.

   In all problems, you may assume that the the dimensions of the matrices and/or vectors are compatible for the requested mathematical operations. **Note**: Throughout the assignment, please use `np.array`, not `np.matrix`.

   (a) Given matrices $\mathbf{A}$ and $\mathbf{B}$, compute and return an expression for $\mathbf{A} + \mathbf{B}$. [**0 pts**]
   Answer: While it is completely valid to use `np.add(A, B)`, this is unnecessarily verbose; you really should make use of the "syntactic sugar" provided by Python's/`numpy`'s operator overloading and just write: `A + B`. Similarly, you should use the more compact (and arguably more elegant) notation for the rest of the questions as well.

   (b) Given matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, return $\mathbf{A} \odot \mathbf{B} + \mathbf{C}^\top$, where $\odot$ represents the element-wise (Hadamard) product and $\top$ represents matrix transpose. Look for highly concise ways of expressing the product and transpose. [**1 pts**]

   (c) Given column vectors $\mathbf{x}$ and $\mathbf{y}$, compute the inner product of $\mathbf{x}$ and $\mathbf{y}$ (i.e., $\mathbf{x}^\top \mathbf{y}$). [**1 pts**]

   (d) Given matrix $\mathbf{A}$ and integer j, return the sum of all the entries in the jth column *whose row index is even*, i.e., $\sum_{i:i \text{ is even}} \mathbf{A}_{ij}$. Do **not** use a loop, which in Python can be very slow. Instead use the `np.sum` function. [**2 pts**]

   (e) Given matrix $\mathbf{A}$ and scalars c, d, compute the arithmetic mean over all entries of A that are between c and d (inclusive). In other words, if $\mathcal{S} = \{(i,j) : c \leq \mathbf{A}_{ij} \leq d\}$, then compute $\frac{1}{|\mathcal{S}|} \sum_{(i,j) \in \mathcal{S}} \mathbf{A}_{ij}$. Use `np.nonzero` along with `np.mean`. [**2 pts**]

   (f) Given a column vector (with n components) $\mathbf{x}$, an integer k, and positive scalars m, s, return an $(n \times k)$ matrix, each of whose columns is a sample from multidimensional Gaussian distribution $\mathcal{N}(\mathbf{x} + m\mathbf{z}, s\mathbf{I})$, where $\mathbf{z}$ is column vector (with n components) containing all ones and $\mathbf{I}$ is the identity matrix. Use either `np.random.multivariate_normal` or `np.random.randn`. [**3 pts**]

   (g) Given a matrix $\mathbf{A}$ with n rows, return a matrix that results from **randomly permuting** the columns (but not the rows) in $\mathbf{A}$. [**2 pts**]

   (h) Z-scoring: Given a vector $\mathbf{x}$, return a vector $\mathbf{y}$ such that each $y_i = (x_i - \overline{\mathbf{x}})/\sigma$, where $\overline{\mathbf{x}}$ is the mean (use `np.mean`) of the elements of $\mathbf{x}$ and σ is the standard deviation (use `np.std`). [**2 pts**]

   (i) Given an n-vector $\mathbf{x}$ and a non-negative integer k, return a $n \times k$ matrix consisting of k copies of $\mathbf{x}$. You can use numpy methods such as `np.newaxis`, `np.atleast_2d`, and/or `np.repeat`. [**2 pts**]

2. **Debugging numpy errors.** For this question ONLY, you can use ChatGPT to debug the code. If you do so, you need to: (i) disclose it in your answer, (ii) indicate if you agree with ChatGPT's initial answer and, if not, (iii) you can describe how you got it to give you the correct answer or you can figure it out by yourself.

   (a) Why the code below isn't subtracting each row's minimum element from the respective row? [**2 pts**]

```
X = np.arange(9).reshape(3,3)
row_min = X.min(axis=1)
print(X-row_min)
```

(b) Change the code above so as to compute an array $\mathbf{Y}$ of shape $(3, 3, 3)$ such that:

$$\mathbf{Y}_{i,j,k} = \mathbf{X}_{j,k} - \texttt{row\_min}_i.$$

You **must not** use loops. Explain the steps performed by numpy's broadcasting. [**2 pts**]

3. **Linear Regression via Analytical Solution**

(a) Train an age regressor that analyzes a $(48 \times 48 = 2304)$-pixel grayscale face image and outputs a real number $\hat{y}$ that estimates how old the person is (in years). Your regressor should be implemented using linear regression. The training and testing data are available here:

- `https://canvas.wpi.edu/files/6219765/download?download_frd=1`
- `https://canvas.wpi.edu/files/6219783/download?download_frd=1`
- `https://canvas.wpi.edu/files/6219715/download?download_frd=1`
- `https://canvas.wpi.edu/files/6219716/download?download_frd=1`

To get started, see the `train_age_regressor` function in `homework1_template.py`.

Note: you must complete this problem using only linear algebraic operations in `numpy` – you may **not** use any off-the-shelf linear regression software, as that would defeat the purpose.

Compute the optimal weights $\mathbf{w} = (w_1, \ldots, w_{2304})$ for a linear regression model by deriving the expression for the gradient of the cost function w.r.t. $\mathbf{w}$, setting it to 0, and then solving. Do **not** solve using gradient descent. The cost function is

$$f_{\text{MSE}}(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} (\hat{y}^{(i)} - y^{(i)})^2$$

where $\hat{y} = g(\mathbf{x}; \mathbf{w}) = \mathbf{x}^\top \mathbf{w}$ and n is the number of examples in the training set.
$\mathcal{D}_{\text{tr}} = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)})\}$, each $\mathbf{x}^{(i)} \in \mathbb{R}^{2304}$ and each $y^{(i)} \in \mathbb{R}$. Note that this simple regression model does not include a bias term (which we will add later); correspondingly, please do **not** include one in your own implementation. After optimizing $\mathbf{w}$ only on the **training set**, compute and report the cost f_{MSE} on the training set $\mathcal{D}_{\text{tr}}$ and (separately) on the testing set $\mathcal{D}_{\text{te}}$. Please report these numbers in the PDF file. [**10 pts**]

4. **Probability Distributions**

(a) **Estimating the Parameters of a Probability Distribution**: Plot the empirical probability distribution of the data in the file `https://canvas.wpi.edu/files/6219717/download?download_frd=1` (you can use `matplotlib.pyplot.hist` with `density=True`). Then, using `scipy.stats.poisson`, plot the probability distributions of a Poisson random variable with (alternative) rate parameters of 2.5, 3.1, 3.7, and 4.3. (Make sure to include these plots in your homework submission.) Based on visual inspection of the idealized distributions with the empirical distribution, report which of the possible parameter values is most consistent with the data. (Later on, we will formalize the parameter estimation process with Maximum Likelihood Estimation.) [**3 pts**]

(b) **Conditional Probability Distributions to Represent the Uncertainty of Functions**: Consider the conditional probability distribution

$$P(y \mid x) = \mathcal{N}\left(\mu = x^2, \sigma^2 = \left(2 - \frac{1}{1 + e^{-x^2}}\right)^2\right)$$

(where $\mathcal{N}$ is the Normal distribution with a specified mean and variance) representing the uncertainty of the y-value associated with any given value x.

i. (Without coding) For which values of x – those with small magnitude, or those with large magnitude – does the corresponding value of y tend to be larger? Explain the steps in your reasoning. [**1 pt**]

ii. (Without coding) For which values of x – those with small magnitude, or those with large magnitude – does the *uncertainty* in the corresponding value of y tend to be larger? Explain the steps in your reasoning. [**1 pt**]

iii. (Using the appropriate scipy function) For $x = 1$, what is the probability that a r.v. Y sampled from that distribution is positive? [**1 pt**]

5. **Proofs/Derivations** For the proofs, please create a PDF (which you can generate using LaTeX, or, if you prefer, a scanned copy of your **legible** handwriting).

   (a) Let $\nabla_{\mathbf{x}} f(\mathbf{x})$ represent the column vector containing all the partial derivatives of f w.r.t. $\mathbf{x}$, i.e.,

   $$\nabla_{\mathbf{x}} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

   For any two column vectors $\mathbf{x}, \mathbf{a} \in \mathbb{R}^n$, prove that

   $$\nabla_{\mathbf{x}} \left(\mathbf{x}^\top \mathbf{a}\right) = \nabla_{\mathbf{x}} \left(\mathbf{a}^\top \mathbf{x}\right) = \mathbf{a}$$

   Hint: differentiate w.r.t. each element of $\mathbf{x}$, and then gather the partial derivatives into a column vector. [**4 pts**]

   (b) Prove that $\nabla_{\mathbf{x}}((\mathbf{x}^\top \mathbf{w} - \mathbf{b})^2) = 2(\mathbf{x}^\top \mathbf{w} - \mathbf{b})\mathbf{w}$ [2 pts]

   (c) Prove that

   $$\nabla_{\mathbf{x}} \left(\mathbf{x}^\top \mathbf{A} \mathbf{x}\right) = (\mathbf{A} + \mathbf{A}^\top)\mathbf{x}$$

   for any column vector $\mathbf{x} \in \mathbb{R}^n$ and any $n \times n$ matrix $\mathbf{A}$. [**4 pts**]

   (d) Based on the theorem above, prove that

   $$\nabla_{\mathbf{x}} \left(\mathbf{x}^\top \mathbf{A} \mathbf{x}\right) = 2\mathbf{A}\mathbf{x}$$

   for any column vector $\mathbf{x} \in \mathbb{R}^n$ and any symmetric $n \times n$ matrix $\mathbf{A}$. [**2 pts**]

   (e) Based on the theorems above, prove that

   $$\nabla_{\mathbf{x}} \left[(\mathbf{A}\mathbf{x} + \mathbf{b})^\top (\mathbf{A}\mathbf{x} + \mathbf{b})\right] = 2\mathbf{A}^\top (\mathbf{A}\mathbf{x} + \mathbf{b})$$

   for any column vector $\mathbf{x} \in \mathbb{R}^n$, any symmetric $n \times n$ matrix $\mathbf{A}$, and any constant column vector $\mathbf{b} \in \mathbb{R}^n$. [**4 pts**]

Submission: Generate (i) a PDF from your answers and (ii) a PDF from your Python code. Combine them into a single PDF and submit on Canvas.
This is an individual homework: You must complete this homework assignment individually.