Assignment 2, FIT5201, S1 2024

Due Date: 11:55PM on 17 May 2024

Version 1.0 (April 30, 2024)

Objectives

This assignment assesses your understanding of

1. Document Clustering,
2. Perceptrons,
3. Neural Networks,
4. Unsupervised Learning
5. Covariances

covered in Modules 4 and 5 in written (Jupyter Notebook) form.

In your Jupyter notebook you can use any import from the libraries `numpy`, `scipy`, `matplotlib`, and `scikit-learn` to solve the tasks except those imports that would render a task trivial. The maximum number of marks of this assignment is 100 (sum of grades from each section). This assignment constitutes 16% of your final mark for this unit. **Make sure you read and understand not only all the tasks but also the notes at the end of this document about submission, assessment criteria, and penalties. Good luck with your assignment!**

1 Document Clustering

Background. In this part, you solve a document clustering problem using unsupervised learning algorithms (Mixture Models with Expectation Maximization) for document clustering. We follow the notations in Module 4. Let $D = \{d_1, d_2, ..., d_N\}$ be a set of N documents and partition them into K clusters. Please use the parameters φ_k and μ_{wk} to represent the prior of the k-th cluster and the occurence probability of the w-th word in the k-th cluster, respectively. All documents share the same vocabulary A.

Question 1 [Expectation Maximisation, 8+4+2+10+8+8=40 Marks]

I Please briefly answer the following high-level questions:

- Write mathematical formulations of the optimization functions of maximum likelihood estimation (MLE) for the document clustering model with complete data and incomplete data, respectively. Then briefly describe why MLE with incomplete data is hard to optimize.
- Briefly explain the high-level idea of the EM algorithm to find MLE parameter estimates.

II Derive Expectation and Maximization steps of the (soft)-EM algorithm for Document Clustering, in a markdown cell (ideally using Latex for clean typesetting and **show your work in your submitted PDF report.**) In particular, include all model parameters that should be learnt and the exact expression (using the same math convention that we saw in the Module 4) that should be used to update these parameters during the learning process (ie., E-step, M-step and assignments).

III Load Task2A.txt file (if needed, perform text preprocessing similar to what we did in Activity 4.2).

IV Implement the EM algorithm (derived in Chapter 5 of Module 4). Please provide enough comments in your submitted code.

Hint: a): If it helps, feel free to base your code on the provided code for EM algorithm for GMM in Activity 4.1. **b):** When implementing the M-step, you will need to normalize the posterior probability matrix in a row-wise manner. Formally, if we need to normalize an N dimensional vector of log probabilities $x_i = \log p_i$ using Softmax, we might naively compute the following equation

$$p_i = \frac{\exp(x_i)}{\sum_{n=1}^{N} \exp(x_n)}, \quad \sum_{n=1}^{N} p_n = 1 \tag{1}$$

Since each x_i is a log probability which may be very large, and either negative or positive, then exponentiating might result in under- or overflow respectively. For example, $\exp(10000) = \text{inf}$. We can simply use the log-sum-exp trick to avoid this numerical unstability. We consider the log-sum-exp operation:

$$\text{LSE}(x_1, \ldots, x_N) = \log\left(\sum_{n=1}^{N} \exp(x_n)\right), \tag{2}$$

$$= c + \log\left(\sum_{n=1}^{N} \exp(x_n - c)\right), \tag{3}$$

where we usually set $c = \max\{x_1, ..., x_N\}$ to ensure that the largest positive exponentiated term is $\exp(0) = 1$, so you will definitely not overflow, and even if you underflow, the answer will be sensible. Then the final equation will be

$$p_i = \exp\left(x_i - c - \log\left(\sum_{n=1}^{N} \exp(x_n - c)\right)\right). \tag{4}$$

You should use this log-sum-exp trick during implementation.

V Set the number of clusters K=4, and run the K-means (hard clustering, using hard-EM) and Mixture Models (soft clustering, using soft-EM) on the provided data. You can reuse the code in Activity 4.2 for K-means.

VI Perform a PCA on the clusterings that you get based on the K-means and Mixture Models in the same way we did in Activity 4.1. Then, plot the obtained clusters with different colors where x and y axes are the first two principal components (similar to Activity 4.2). Based on your plots, discuss how and why the hard and soft clustering are different in a markdown cell.

2 Perceptron vs Neural Networks

Background. In this part, you will be working on a binary classification task on a given synthetic dataset. You will use machine learning tools including a Perceptron and a 3-layer Neural Network to solve the task. Here, we are looking for your meaningful observations and discussions towards the differences between Perceptron and Neural Networks.

Question 2 [Neural Network's Decision Boundary, 2+7+10+6=25 Marks]

I Load Task2B_train.csv and Task2B_test.csv datasets, plot the training and testing data separately in two plots. Mark the data with different labels in different colors.

II Train two Perceptron models on the loaded training data by setting the learning rates η to 0.1 and 1.0 respectively. Calculate the test errors of two models and find the best η and its corresponding model, then plot the decision boundary and the test data in one plot. **Hint:** We expect the decision boundary of your perceptron to be a linear function that separates the testing data into two parts. You may also choose to change the labels from [0, 1] to [-1, +1] for your convenience.

III For each combination of K (i.e, number of units in the hidden layer) in 5, 10, 15, ..., 40, (i.e. from 5 to 40 with a step size of 5), and η (i.e., learning rate) in 0.01, 0.001 run the 3-layer Neural Network and record testing error for each of them. Plot the effect of different K values on the accuracy of the testing data. Based on this plot, find the best combination of K and η and obtain your best model, then plot the decision boundary and the test data in one plot.

IV Explain the reason(s) responsible for such difference between Perceptron and a 3-layer Neural Network by comparing the plots you generated in Steps II and III. **Hint:** Look at the plots and think about the model assumptions.

3 Unsupervised Learning

Background. In this part, you will implement self-taught learning using an Autoencoder and a 3-layer Neural Network to solve a multi-class classification task on real-world data.

Question 3 [Self-supervised Neural Network Learning, 2+5+2+5+5+6=25 Marks]

I Load Task2C_labeled.csv, Task2C_unlabeled.csv, and Task2C _test.csv datasets, along with the required libraries. Note that we will use both Task2C_labeled.csv and Task2C_unlabeled.csv to train the autoencoder, and only Task2C_labeled.csv to train the classifiers. Finally, we will evaluate the trained classifier on the test dataset Task2C_test.csv.

II Train an autoencoder with only one hidden layer and change the number of its neurons to 20, 60, 100, ..., 220 (i.e. from 20 to 220 with a step size of 40).

III For each model in Step II, calculate and record the reconstruction error for the autoencoder, which is simply the average of Euclidean distances between the input and output of the autoencoder. Plot these values where the x-axis is the number of units in the middle layer and the y-axis is the reconstruction error. Then, explain your findings based on the plot.

IV Build the 3-layer NN to build a classification model using all the original attributes from the training set and change the number of its neurons to 20, 60, 100, ..., 220 (i.e. from 20 to 220 with a step size of 40). For each model, calculate and record the test error.

V Build augmented self-taught networks using the models learnt in Step II. For each model:

1) Add the output of the middle layer of an autoencoder as extra features to the original feature set;
2) Train a new 3-layer Neural Network using all features (original + extra) and varying the number of hidden neurons (like Step IV) as well.
3) Then calculate and record the test error.

For example, each model should be developed as follows: Model 1: 20 hidden neurons + extra 20 features (from an autoencoder), Model 2: 60 hidden neurons + extra 60 features (from an autoencoder), ..., Model 5: 220 hidden neurons + extra 220 features (from an autoencoder).

VI Plot the error rates for the 3-layer neural networks from Step IV and the augmented self-taught networks from Step V, while the x-axis is the number of hidden neurons and y-axis is the classification error. Explain how the performance of the 3-layer neural networks and the augmented self-taught networks is different and why they are different or why they are not different, based on the plot.

4 Covariances

Question 4 [Portfolio Risk, 5+5=10 Marks] Assume we have four financial assets and their values (per dollar) in 20 years from today are the random variables X_1, X_2, X_3, and X_4. Assume further that their mean values and standard deviations are $100, 100, 50, 50$ and $20, 22, 5, 5$, respectively. Finally, assume that the correlation matrix between the variables is

$$R = \begin{pmatrix} 1.0 & 0.9 & 0.0 & 0.0 \\ 0.9 & 1.0 & -0.9 & -0.9 \\ 0.0 & -0.9 & 1.0 & 0.5 \\ 0.0 & -0.9 & 0.5 & 1.0 \end{pmatrix}$$

where the entry $R_{i,j}$ corrsponds to $\rho(X_i, X_j) = \text{Cov}(X_i, X_j)/(\text{Std}(X_i)\text{Std}(X_j))$. Now we consider the following four investment portfolios, each investing 100 dollars today:

1. Investing 50 into asset 1, 0 into asset 2, 25 into asset 3, and 25 into assset 4
2. 0, 50, 50, 0
3. 0, 50, 25, 25
4. 25, 25, 25, 25

I Compute the expected values and standard deviations of each portfolio after 20 years using Python (with only imports from `numpy`) and give a recommendation in a markdown cell on what portfolio a conservative investor should choose.

II Explain the mathematical background of your computations in another markdown cell, i.e., explain what rules of probability you applied and how (make sure to mention arguments for both the expected value as well as the standard deviations of the portfolios).

Submission

The assignment is assessed based on file(s) submitted via Moodle. Not submitting the files will result in 0 marks for the assignment.

Submission Please submit one zip-file named according to the schema

`STUD-ID_FIRSTNAME_LASTNAME_assignment2.zip`

that contains two versions (the actual "ipynb"-file and a "pdf"-export) of five Jupyter Notebooks—one for each section of this assignment ("1 Document Clustering", "2 Perceptron vs Neural Networks", etc.). The files contained in the zip file should be named as follows:

`STUD-ID_FIRSTNAME_LASTNAME_a2_sec1.ipynb`
`STUD-ID_FIRSTNAME_LASTNAME_a2_sec1.pdf`
`STUD-ID_FIRSTNAME_LASTNAME_a2_sec2.ipynb`
`STUD-ID_FIRSTNAME_LASTNAME_a2_sec2.pdf`
...

Overall, there should be eight files within the zip-file corresponding to the four sections of the assignment. Furthermore, make sure that notebooks for each question contain

- your name and student ID in a leading markdown cell followed by
- a structure that clearly separates between questions (with markdown headlines and sub-headlines) and then for each question
- all required code,
- all required figures,
- and your answers to all question parts that require a free-text answer (in markdown cells).

Note that depending on your system it might be necessary to first generate an html export and then save that with your web browser as pdf-file. The submission must be received via Moodle by the due date mentioned on the top of this document.

Notes Please note that,

1. One second delay will be penalized as one day delay. So please submit your assignment in advance (considering the possible Internet delay) and do not wait until the last minute.
2. We will not accept any resubmitted version. So please double check your assignment before the submission.

Assessment Criteria

Your final mark for the assignment is sum of marks for your submitted files.

The following outlines the criteria which your **submitted notebooks** will be assessed against:

1. Working code: The code executes without errors and produces correct results.
2. Understandable code: Usage of clear ideoms, function and variable names, and comments where useful.
3. Quality of your written answers: Completeness and logical coherence of arguments, usage of proper technical terms, citation of external sources when used.
4. Marks can be deducted if relevant information is hard to access because of the inclusion of irrelevant and redundant information. For your information: The overall marking time for an individual assignment is set to 20 minutes. If the marker cannot make sufficient sense of your submission during that timefrime they will typically deduct marks.

Penalties

- Late submission (students who submit an assessment task after the due date will receive a late-penalty of 10% of the available marks in that task per calendar day. Assessment submitted more than 7 calendar days after the due date will receive a mark of zero (0) for that assessment task.)
- Any file not properly named (-5%)
- Lack of structure in Jupyter Notebook, e.g., no section title for questions (-5%)