Department of Mathematics
MATH96007 - MATH97019 - MATH97097
Methods for Data Science
Years 3/4/5

Coursework 3 – Unsupervised and supervised learning on high dimensional data

Deadline: Friday, 10 January 2020, 5 pm

General instructions

The goal of this course is to analyse datasets using some of the tools introduced in the lectures and additional tools you will find in the literature following your own initiative. As discussed throughout, coursework tasks are different from exams: they can be more open-ended and may require going beyond what we have covered explicitly in lectures. Initiative and creativity beyond the strict statements of the task are important, as is the ability to pull together the mathematical content of the course, drawing new links between subjects and methods, and backing up your analysis with relevant computations. The quality of presentation and communication is very important, so use good combinations of tables and figures to present your results. Explanation and understanding of the mathematical concepts are crucial, and competent code is expected.

In this coursework you must submit **three files:**

(1) You must **submit a Jupyter notebook** with all your tasks. The notebook should contain text explaining your calculations, interspersed with the corresponding cells containing the code and the corresponding output. The notebook should run to produce the outputs. You can produce your notebook within Google Colab, or develop your local Jupyter notebook through the Anaconda environment downloaded on your computer.

> ***Note:*** *If you are expecting to travel over the holidays, and plan to work while away, and foresee any problems with connecting to Google Colab from your travel destination, you must make sure that you have downloaded and installed Anaconda and all the necessary packages on your own computer and that they are functional so that you can work on your own computer.*

(2) When your notebook is complete, you should execute the notebook and you will need to **submit the html file** produced from it. Before you produce your html, make sure all cells are executed, so the complete output appears in your html file.

(3) In this coursework, you will also submit the **pdf file of a poster** summarising one part of your results.

Instructions and guidance about solutions and marking scheme can be found at the end of this document.

Marks

The coursework is worth **50% of your total mark**, and contains a **mastery component** for MSc and MSci students.

Dates

The coursework is due on **10 January 2020**, i.e., at the end of the first week of Term 2.

There will be two computer sessions at the MLC to answer your questions on:

- 12 December 2019 from 2-4 pm
- 9 January 2020 from 2-4 pm.

I will also hold my usual office hours on:

- 6 January 2020 from 12-1 pm.

*As per the Course Outline discussed at the beginning of the term, **there will be no out-of-term assistance with the coursework by myself or the GTAs**. We will all be travelling and off the grid for those weeks, and we will not be checking or responding to email. Have a good break and best wishes for 2020.*

Overview

In this final coursework, you will work with two very different datasets of high-dimensional samples: **a dataset of text documents** and **a collection of images**. Your tasks will range from *unsupervised learning to graph analytics to supervised classification to dimensionality reduction*.

Task 1: Unsupervised learning: text documents with an associated citation graph (45 marks)

Dataset: Your first dataset is a collection of text documents. Each sample in the dataset corresponds to a journal paper in some scientific (sub)discipline. The text in each paper has been summarised as a high-dimensional vector of *p* features, where each (binary) coordinate of the vector indicates the presence or absence of a particular word within the text of the paper. The size of the dictionary for this dataset is 1433 words. Hence the dataset contains *N*=2485 papers, and each paper is described by a *p*-dimensional vector of features (*p*=1433). This dataset is given as an *N x p* feature matrix, *F.*

For this dataset we have also information about citations between papers. The citations define an undirected graph (we ignore directionality of citations here) which is encoded through its *N x N* adjacency matrix, *A*.

Both the text feature matrix *F* and the adjacency matrix *A* of the citation graph are available on Blackboard.

Description of the task: This task focuses on exploring the structure of this dataset through unsupervised learning both on the features and, separately, on the associated graph. You will attempt to learn about clusters of documents that have similar content and compare such clustering to groups of documents that cite each other obtained from the citation graph. You will also characterise the importance of documents according to the graph citation structure.

> ***Note:*** *To carry out this task, you will use sklearn and the NetworkX package within Python. The NetworkX package contains pre-implemented graph-theoretical algorithms and good plotting capabilities for networks that you should use to report your results.*

Questions:

1.1 Clustering of the feature matrix (15 marks): Find optimised clusterings of the feature matrix *F* by running the k-means algorithm for all values of *k* in the interval [2, 30]. Plot the Calinski-Harabasz (CH) score to evaluate the quality of clusterings as a function of *k,* and pick the first clustering with CH < 7 as your 'optimal' clustering.

Report some of the characteristics of this 'optimal' clustering (e.g., distribution of cluster sizes; within group and across group similarities; or any other relevant statistics or descriptors of the clustering at your discretion).

Discuss the meaning of the CH score and evaluate the quality of the 'optimal' clustering you found according to other quality measures from the literature.

Evaluate the robustness of your results given the randomness of k-means optimisation.

1.2 Analysis of the citation graph (10 marks): Use NetworkX to display the citation graph described by the adjacency matrix, *A*. Plot the degree distribution of the graph as a histogram.

Compute the following three centrality measures for all the nodes of the graph: (i) degree; (ii) betweenness centrality; (iii) Pagerank. Study which nodes (if any) are highly central according to the three measures.

Using appropriate correlation plots (or otherwise), discuss the similarity between the node rankings according to the different centrality measures and explain why the centrality rankings might differ.

1.3 Community detection on the citation graph (10 marks): Use the Clauset-Newman-Moore greedy modularity maximisation algorithm in Networkx to compute the optimal number of communities *k** and the corresponding partition of the citation graph.

Plot the obtained clusters on the NetworkX graph you obtained in 1.2 by assigning different colours to the nodes in each community.

Study how the top 30 most central nodes according to degree and Pagerank computed in 1.2 are distributed across your *k** communities. Explain your findings.

1.4 Compare feature and graph clusterings (10 marks): Use Adjusted Mutual Information (AMI) and Adjusted Rand Index (ARI) to score how similar the optimal clusterings obtained in 1.1 and 1.3 are to each other.

Plot the clusters you obtained in 1.1 onto the NetworkX graph you generated in 1.2 so that you can compare visually the feature clusters in 1.1 to the graph-based communities in 1.3.

Based on the above, discuss the similarities and differences you observe. You can use other metrics or other visualisation methods to compare the two optimal clusterings at your discretion, explaining your choices.

> ***Note:*** *There is a 'ground truth' number of categories for this dataset (Cora), which you may find online. However, the point of this task is* ***not*** *to discover those 'ground truth' categories (which, incidentally, are not clearly accepted), but instead to reveal the intrinsic structure in the data directly from both features and graph.*

Task 2: Classification of a set of images (45 marks)

Dataset: Your second dataset is a collection of images of fashion items sold by Zalando. This dataset is commonly referred to as Fashion-MNIST. Each sample is originally a grayscale image with 28x28 pixels. Each pixel has a value of the grayscale between 0 and 255. In some of the tasks below, the images will need to be described as *p*-dimensional vectors ($p=28^2$); in other tasks, the images will need be considered as 28x28 matrices.

Each image in the dataset belongs to one of C=10 classes with the following labels:

Label	0	1	2	3	4	5	6	7	8	9
Description	T-shirt/top	Trousers	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boots

The dataset consists of 60,000 images in the training set and 10,000 additional images in the test set. As you know, the test set should **not** be used in any learning and should only be used *a posteriori* to support your conclusions and to evaluate your models.

The dataset is available through PyTorch and sklearn so that you do not have to download it separately, and you can use it directly from your Python code.

Hint: *Within sklearn you can use the fech_openml package to get the dataset:*

```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('Fashion-MNIST', cache=False)
X = mnist.data.astype('float32')
y = mnist.target.astype('int64')
```

As a safety, we have also made the dataset available on Blackboard.

Description of the task: This task has two aspects. First, you will explore the dataset in an unsupervised manner and evaluate how clearly separated the classes are. Second, you will switch to supervised learning and use some of the ground truth labels to train two neural networks with different architectures for supervised classification.

Questions:

2.1 Unsupervised clustering of the image dataset (20 marks): Use the *k*-means algorithm in sklearn in an unsupervised manner (i.e., without using any 'ground truth' labels) to cluster the Fashion-MNIST dataset (*just the 'training' part*) into *k* classes, for all values of *k* between *k*=2 and *k*=30. Give some thought to how to deal with the inherent randomness of the output of the k-means algorithm.

Do you find any evidence of the existence of 10 classes in the data? Support your explanation using some of the measures you discussed in Task 1.1, or otherwise using *other* unsupervised clustering methods described in class. Support your explanations with figures, computations and mathematical explanations.

Consider the k-means clustering you obtained for k=10. Visualise the centroid of each of the 10 clusters.

Use the k-means clustering with k=10 as a **kNN classifier for the test set**. Report its accuracy and any other quality measures of its performance.

2.2 Supervised classification of the training set (25 marks)

2.2.1 MLP neural network supervised classification: Using PyTorch, implement a multilayer perceptron (MLP) to classify the images of the dataset into the 10 classes. For this task, you will need to vectorise your images so that they can be used as inputs.

Setup of the network: Your MLP will have a number of input nodes equal to the number of pixels in each image, three hidden layers, each with 100 nodes, and an output layer with 10 nodes corresponding to the number of classes. You should use ReLU as your activation function with no dropout.

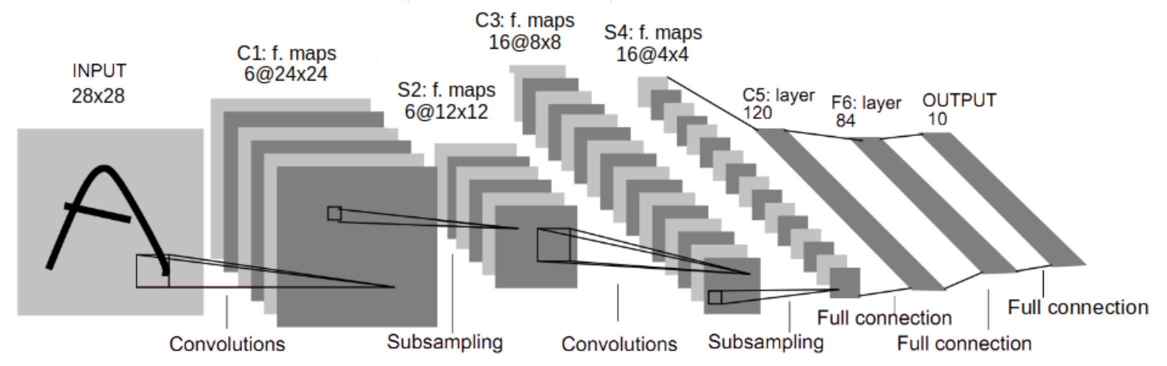
For the training, you should use a negative log-likelihood loss, a learning rate of 0.005, and stochastic gradient descent (SGD) as the optimisation method. You should train in batches of 128 images for 30 epochs.

Note: *If using Google Colab, make sure you use its GPUs by selecting in the top menu 'Edit > Notebook settings > Hardware accelerator > GPU'.*

Compute the accuracy of this MLP classifier **on the test set** and report any other quality measures of its performance. No optimisation of hyperparameters is necessary.

2.2.2 Convolutional neural network (CNN) supervised classification: Using PyTorch, build a small CNN to classify the images of the dataset into the 10 classes.

<u>*Setup of the network:*</u> Your CNN should have the following architecture:



You should choose ReLU as the activation functions, reduce the dimensionality of the feature maps by a sub-sampling layer where the highest value in each subsampled feature map is chosen, implement a stride of 2 into these pooling layers, and have a kernel size of 5 in all convolutional layers. The stride and the dilation in the convolutional layers should be both 1. All other hyperparameters and options remain as in 2.2.1.

***Note:** If using Google Colab, make sure you use its GPUs by selecting in the top menu 'Edit > Notebook settings > Hardware accelerator > GPU'.*

Compute the accuracy of this CNN classifier **on the test set** and report any other quality measures of its performance.

2.2.3 Comparisons of the classifiers: Compare the performance of the CNN and MLP classifiers focussing in particular on the difference in the number of parameters in both models. Explain the observed difference in performance given the different operations on the data carried out by the MLP and CNN models.

Compare the performance of both supervised classifiers to the k-NN classifier obtained in 2.1 from the unsupervised k-means clustering. Discuss the outcomes given the distinct approaches used in supervised and unsupervised learning.

Using 5-fold cross-validation on the training set, optimise the CNN architecture to increase its accuracy above 90%. You might want to try adding hidden layers, changing the activation function, or any other method you believe could increase the accuracy. Explain your choices and why these changes result in an improvement.

Task 3: Poster (10 marks)

Prepare a **pdf file** with a poster to illustrate your work and findings for **either** Task 1 **or** Task 2. You are free to choose the format of your poster. The poster should contain a concise description of the data and the problem; a summary of the methods used, and a compilation of your findings and conclusions.

The poster should be concise but clear; should not contain large amounts of text; and use clearly labelled figures and tables obtained with your Python scripts. Keep to message but be clear about your conclusions.

Your target audience would be, e.g., a manager working for Nature Publishing Group (for Task 1) or a manager working for Zara or Uniqlo (for Task 2), looking to hire a data scientist to join their company. There is more guidance on the poster at the end of the Coursework.

Task 4: Dimensionality reduction of images *(for MSc and MSci students only)* (25 marks)

Description of task: This last task explores the use of dimensionality reduction to describe the Fashion-MNIST image dataset in Task 2 in an unsupervised manner. You are expected to read the published literature (e.g.,, the original paper on NMF) to complete your answers beyond what we explained in lecture.

4.1 Comparing PCA and NMF dimensionality reduction: Use principal component analysis (PCA) and non-negative matrix factorisation (NMF) to reduce the dimensionality of the Fashion-MNIST dataset (just the 'training' part) to the top 10 'components'. For each of the two methods, visualise each of the 10 components and study quantitatively how they are related to the 10 classes in the data. Use your computations to explain the differences between the PCA and NMF representations in terms of the sparsity of the corresponding components and the correspondence of the components to visual features (see NMF original paper by Lee and Seung).

4.2 Latent Dirichlet Allocation (LDA) applied to images: Latent Dirichlet allocation (LDA) is a method introduced by Blei, Ng and Jordan, which is widely used in Natural Language Processing (NLP) to analyse collections of text documents (like the one you were given in Task 1) to group documents into 'topics' according to their similarity. You should read the description of LDA in the literature.

Making the following analogies (document→image, word→pixel), apply LDA to the Fashion-MNIST dataset (just the 'training' part) in order to find 10 'topics' corresponding to groups of images that display high similarities.

Visualise each of the centroids of the 10 groups you found using LDA and compare them with those found in 4.1 and 2.1. Would you expect LDA to work on this problem? Give some reasons based on your reading of the method.

General guidance:

Guidance about solutions and marking scheme:

To gain the marks for each of the Tasks you are required to: (1) complete the task as described; (2) comment any code so that we can understand each step; (3) provide a brief written introduction to the task explaining what you did and why you did it; (4) provide appropriate, relevant, clearly labelled figures documenting and summarising your findings; (5) provide an explanation of your findings in *mathematical terms* based on your own computations and analysis and linking the outcomes to concepts presented in class or in the literature; (6) consider summarising your results of different methods and options with a judicious use of summary tables.

Marks will also be reserved and allocated for: presentation; quality of code; clarity of arguments; explanation of choices made and alternatives considered; mathematical interpretation of the results obtained; as well as additional *relevant* work that shows initiative and *understanding* beyond the minimal task stated in the coursework. However, the mere addition of extra calculations (or ready-made 'pipelines') that are unrelated to the task without a clear explanation and justification of your rationale will not be beneficial and, in fact, can also reveal lack of understanding.

Specific guidance for the poster:

Your poster should tell the story of your data science project. Imagine showing it to a potential employer or tweeting about it, or presenting it a workshop. Your reader should be able to understand the data you had, the question you asked, how you answered it, and what you found from the analysis.
Avoid using long chunks of text in paragraphs: the poster should look good and be accessible to read. Of course, you will need to use *some* text to explain what you did and why, but try to keep the text concise and clear. Bullet points, or short sentences are typically more effective than long text paragraphs.
You can create your poster with any software of choice, but you are encouraged to use LaTeX. There are poster templates at www.overleaf.com which you can use without even having a LaTeX installation on your computer.

Keep in mind:

- The poster should be written clearly and should communicate your project's story with correct spelling and grammar. Plots must have titles and axis labels with legible font size. Unlabelled plots may cause you to lose marks.
- Because of its format, it is impossible to include every detail in a poster. This is OK: a key to good communication is having the 1-minute, 3-minute, 10- minute versions of your story. Think about how to describe your project in three sentences: what you asked, what you did, something you found.
- All figures in your poster must be produced in your notebook.

The poster will be marked for clarity, completeness, and accuracy of mathematical and methodological presentation, as well as for originality and general impression.

Submission instructions

You will upload **three documents** to Blackboard, wrapped into a single zip file:

1) Your notebook as an ipynb file.
2) Your notebook exported as an html file.
3) Your poster as a pdf file.

You are also required to comply with these specific requirements:

- Name your files as 'SurnameCID.zip', e.g. Smith1234567.zip. Do not submit multiple files.
- Your ipynb file must produce all plots that appear in your html file, i.e., make sure you have run all cells in the notebook before exporting the html.

- Your poster should only contain figures that you have produced in your notebook.
- Use clear headings in your html to indicate the answers to each question, e.g. ‘Task 1.1’.

Note: *There seem to be some issues with particular browsers (or settings with cookies or popup blockers) when submitting to Turnitin. If the submission 'hangs', please try another browser. You should also check that your files are not empty or corrupted after submission.*

To avoid last minute problems with your online submission, we recommend that you upload versions of your coursework early, before the deadline. You will be able to update your CW until the deadline but this provides you with some safety back up.

Needless to say, projects must be your own work: *You may discuss the analysis with your colleagues but the code, writing, figures and analysis must be your own. The Department may use code profiling and tools such as Turnitin to check for plagiarism, as plagiarism cannot be tolerated.*