Topic	Description
Module	Data Science II
Module Code	MECH-B-5-MLDS-MLDS2-ILV
Semester	WS 2025
Lecturer	Daniel T. McGuiness, Ph.D
ECTS	5
SWS	4
Lecture Type	ILV
Coursework Name	Individual Assignment
Work	Individual
Suggested Private Study	10 hours
Submission Format	Submission via SAKAI
Submission Deadline	20.01.26 17:00
Late Submission	Not accepted
Resubmitting Opportunity	No re submission opportunity

No lecture time is exclusively devoted to the aforementioned assignment.

This is an individual work and collaborative work will **NOT** be accepted.

This assignment is composed of three (3) questions on machine learning methods which you are to solve using Python and the write the report in LaTeX. For any questions, please send your emails to:

Daniel.McGuiness@mci.edu

A portion of the mark for every assignment will be, where applicable, based on style. Style, in this context, refers to organisation, flow, sentence and paragraph structure, typographical accuracy, grammar, spelling, clarity of expression and use of correct IEEE style for citations and references. Students will find *The Elements of Style (3rd ed.)* (1979) by Strunk & White, published by Macmillan, useful with an alternative recommendation being *Economist Style Guide (12th ed.)* by Ann Wroe.

Submission Requirements with Python

Before submitting your work for assessment please read the following requirements.

1. Work submitted using iPython notebooks (`.ipynb`) will be **rejected**.
2. Only work submitted to SAKAI will be considered. Links to private repos and email will **NOT** be considered for grading.
3. Code should be single spaced and use readable font sizes. Avoid using Tabbing.
4. Add spacing around operators if it improves clarity.

> To have a good overview of industrial standards of python programming, please have a look at PEP 8 - Style Guide for Python Code and NumPy Documentation Style Guide.

Commenting

1. The goal of commenting is to make your code easily understandable by someone else.
2. There should be a block comment at the start of your program explaining what it does and including: your name, the assignment and the date.
3. Every function/method should have a block comment explaining what is does.
4. Add inline comments where needed to explain subtle or tricky code.
5. Add blank lines and comments to separate sections of large methods that perform several different tasks.

Naming Variables, Methods, and Classes

1. Take the time to choose meaningful names.
2. Follow naming conventions (i.e., PEP, NumPy Docs).
3. You can use abbreviations, but add comments explaining them.

> Many people comment out old code that wasn't working correctly. Clean up and remove any old code before turning it in.
>
> **If there is any code which is left that contributes no functionality to the operation or the documentation of the code, points will be deducted from the final grade.**

> If there is any suspicion of the submitted code being written using external aids (i.e., AI, web-references) without explicit declaration of exactly where it was used, the code shall be deemed **plagiarised**.

An Example Code with Documentation

An example documentation which is the required standard for the submitted work is as follows:

```
class ExampleClass(object):
    """The summary line for a class docstring should fit on one line.

    If the class has public attributes, they may be documented here
    in an ``Attributes`` section and follow the same formatting as a
    function's ``Args`` section. Alternatively, attributes may be documented
    inline with the attribute's declaration (see __init__ method below).

    Attributes
    ----------
    attr1 : str
        Description of `attr1`.
    attr2 : :obj:`int`, optional
        Description of `attr2`.
    """

    def __init__(self, param1, param2, param3):
        """Example of docstring on the __init__ method.

        The __init__ method may be documented in either the class level
        docstring, or as a docstring on the __init__ method itself.

        Either form is acceptable, but the two should not be mixed. Choose one
        convention to document the __init__ method and be consistent with it.

        Note
        ----
        Do not include the `self` parameter in the ``Parameters`` section.

        Parameters
        ----------
        param1 : str
            Description of `param1`.
        param2 : :obj:`list` of :obj:`str`
            Description of `param2`. Multiple
            lines are supported.
        param3 : :obj:`int`, optional
            Description of `param3`.
        """
        self.attr1 = param1
        self.attr2 = param2
        self.attr3 = param3  #: Doc comment *inline* with attribute

        #: list of str: Doc comment *before* attribute, with type specified
        self.attr4 = ["attr4"]

        self.attr5 = None
        """str: Docstring *after* attribute, with type specified."""

```

```
50      def example_method(self, param1, param2):
51          """Class methods are similar to regular functions and documented the same.
52
53          Note
54          ----
55          Do not include the `self` parameter in the ``Parameters`` section.
56
57          Parameters
58          ----------
59          param1
60              The first parameter.
61          param2
62              The second parameter.
63
64          Returns
65          -------
66          bool
67              True if successful, False otherwise.
68
69          """
70          return True
```

Submission Format and Requirements with LaTeX

Before submitting your work for assessment please follow the following requirements.

1. **The work must be done using LaTeX and no other format will be accepted.** You are allowed to use whatever software you wish to write (i.e., Overleaf, TeXStudio, LyX, ...). The work must also conform to the **MCI** Documentation guidelines (which is in SAKAI). A template is provided to you here which was written by the author of this document.

   > If you are to use **mcidoc** template, set the `document-state` as `Report`.

   > To learn LaTeX, the author has tutorials which can be accessed here.

   Please treat this assignment as a great practice to learn LaTeX as you will need it when you are writing your thesis.

2. The report (and all its content) **must** be submitted as a `.pdf` along with all the necessary files (`.tex`, `.sty`) to compile it. Submit one (1) `.zip` file containing everything with the following name pattern.

   `[STUDENT-ID]_[STUDENT-NAME]_[STUDENT-SURNAME].zip`

   For example, a student named **Jason Brigham** with a student ID **2710704051** would send their submission as:

   `2710704051_Jason_Brigham.zip`

   and their submission would be in arranged as the following directory:

   ```
   2710704051_Jason_Brigham/
   ├── tex/
   │   ├── 2710704051_Jason_Brigham.tex, *.pdf, mcidoc.cls, custom.sty, ...
   │   └── figures/
   │       └── .jpg, .png, ...
   └── HW_Content/
       └── src/, include/, .cpp, .bash, .py, ...
   ```

   where `...` are any additional requirement (i.e., `.bib`, `.glo`) you may need for TeX and the `HW_Content` is any content which is relevant to the Homework (i.e., if your HW is about python, this is where all the python scripts would be kept, or where your ROS 2 source files would be, or where your C++ codes would be for a project.)

3. The work must be submitted to SAKAI and **no link to personal repo will be accepted.**

> To make sure your submission conform to given requirements, please test out zipping/unzipping and running any code before submission. Any deviations from the requirements will incur penalties to the final grade.

Question	Maximum Point	Received Point
Visualising Olivetti Faces	40	
Clustering the Faces	30	
Gaussian Mixture Models	30	
Sum	100	

[Q1] Visualising Olivetti Faces — 40

Figure 1: The Olivetti dataset.

The classic Olivetti faces data set contains 400 gray-scale 64×64 pixel images of faces. Each image is flattened to a 1D vector of size 4,096. Forty different people were photographed, with 10 times each, and generally the task is to train a model which can predict which person is represented in each picture, shown in **Fig.** 1.

Load the dataset using the `sklearn.datasets.fetch_olivetti_faces()` function, then split it into a training set, a validation set, and a test set.

> The dataset is already scaled between 0 and 1.

Given the dataset is quite small, a suggestion is to use stratified sampling to ensure there are the same number of images per person in each set. Following this, cluster the images using k-means, and ensure you have a good number of clusters, and finally visualise the clusters.

[Q2] Clustering the Faces — 30

Continuing with the previous question, using the same dataset, now train a classifier to predict which person is represented in each picture, and evaluate it on the validation set. Next, use *k*-means as a dimensionality reduction tool, and train a classifier on the reduced set.

Search for the number of clusters which allows the classifier to get the best performance and determine what performance can you reach.

[Q3] Gaussian Mixture Models — 30

Finally, train a Gaussian mixture model on the same dataset.

> To speed up the algorithm, a suggestion would be to reduce the datasets dimensionality.

Use the model you made to generate some new faces (using the `sample()` method), and visualise them.

Try to modify some images (e.g., rotate, flip, darken) and see if the model can detect the anomalies and review your results.

Report You are to write a report on your individual assignment which should explain the methods used in solving the given questions. A template structure for this assignment could be as follows. Per each question:

- An introduction to the question, and the method used to tackle.
- A light literature review (2-3 sources) on the method used in tackling the given problem.
- Explanation (i.e, conceptual and mathematical) of the method(s) used in the tackling of the problem, such as the code used along with its proper documentation.

  For documentation of a function or class, use numpy style documentation.
- Per each question there should be one (1) code file (`HW1.py`, `HW2.py`, ...) which should be place in your `HW_Content` folder.
- Presenting of its results and discussion.

> Before submission, please read **Submission Format and Requirements with LaTeX** and Submission requirements with Python.