Foundations of Data Analysis - WS22
Lab assignment
Supervised learning

Due date: 09:45 am on 30.11.2022

Description and Instructions

The maximum number of points achievable in this assignment is 100. Please follow the submission instructions carefully, as failing to do so can **result in a penalty**.

- You should work on this assignment individually, but you are allowed and encouraged to discuss your approaches to the problems, as well as questions you may have, with fellow students.
- You are, however, not allowed to share your code! (Except for very small parts, in order to discuss problems with your peers.)
- Remember to cite every external source that you use (as comments in your code)!
- Any act of plagiarism will be taken very seriously and handled according to university guidelines.

Do not hesitate to email me (Christoph Luther) at `christoph.luther@univie.ac.at` or post on the discussion forum on Moodle with any questions you may have.

Introduction

The purpose of this assignment is for you to put the theory about supervised learning algorithms we have discussed in the lectures into practice. In the first task, you are given a synthetic data set on which you have to perform linear discriminant analysis (**lda**) while going through several steps necessary when you apply prediction models in practice. In the second task, you are given a data set of images, and we ask you to try your best to train an image classifier (**img**). You are free to use any machine learning method you like, both those presented in the course and other methods, as well as any data preprocessing you think can improve your model. We specifically encourage you to do some research on which methods might be particularly suited for this kind of data, play around with your models, and to **use methods beyond those covered in class**. In short: Give this challenge all you can come up with!

Formal Requirements

Download the following files from u:cloud.
Link: https://ucloud.univie.ac.at/index.php/s/bPXvGx5gzJrLijE ;
Password: `fda_ws22)BZ/=fej`

- `lda_data.csv` (task 1: **lda**)
- `lda_template.py` (task 1: **lda**)
- `X_train.csv` (task 2: **img**)
- `y_train.csv` (task 2: **img**)
- `img_template.py` (task 2: **img**)
- `requirements.txt` (both tasks)

`lda_data.csv` contains the synthetic data for the **lda** task and `lda_template.py` a corresponding template that you **can** use if you want to.

`X_train.csv` contains flattened images of dimensions $28 \times 28 \times 1$, `y_train.csv` contains the corresponding labels (94 different characters). `img_template.py` is the corresponding template that you **can** use if you want to. Note, however, that you are required to provide a function `train_predict(X_train,y_train,X_test)` which returns a prediction `y_pred` as can be found in the template as part of your solution. Also, import the data according to the template and adhere to the instructions therein. Hence, we highly recommend to work within the template. You should add additional functions as needed at the top of the file, and adapt the provided function to include your data preprocessing, model definition, training and prediction where indicated. The template also contains checks of the input and output formats. To evaluate your model we will import the function `train_predict` and call it on the provided training data `X_train` and `y_train` and a secret test data `X_test`. The returned predictions `y_pred` will be compared with the secret `y_test` to compute the accuracy of your model which will make up part of your final score.

We will execute your code in an environment created from the file `requirements.txt`[1]. We therefore strongly recommend that you recreate this environment locally. To do so, you can create a virtual environment according to venv. Once you activated the environment, install all necessary packages from `requirements.txt`, e.g. by executing **pip install -r requirements.txt**. However, you can use similar solutions, like conda environments (conda), too[2]. The environment contains the packages discussed in the lecture and beyond. You can use the information on the packages as hint for your solution and we ask that you complete the assignment with only those. If you absolutely want to use additional packages, contact me to ask permission.

To submit your solutions, upload two python files to Moodle - one for each task, named `<last_name>_<letter_first_name>_<task>.py`, replacing `<last_name>` with your last name(s), `<letter_first_name>` with the first letter of your first name and `<task>` with

[1] In Python Version 3.9.12
[2] e.g. `conda create --name fda python=3.9.12` + installing packages

either `lda` or `img` indicating the task. For task 1 (**lda**), however, we also accept Jupyter notebooks. (Example: `luther_c_lda.py`/`luther_c_lda.ipynb` and `luther_c_img.py`).

Hints:

- It may be easier to develop your models in the console or a jupyter notebook, and only later add your final processing pipelines to the templates for task 2.
- Make sure to set aside part of the training data as validation set, in order to estimate the performance of your model on unseen data!

1 Linear Discriminant Analysis (lda)

In this task you are guided through the application of linear discriminant analysis. You have to perform the steps on the data set `lda_data.csv`. The data set consists of three columns labelled "$X1$", "$X2$" and "Y", where "$X1$" and "$X2$" contain observations for two features that you are supposed to use to train a classifier for the binary label "Y".

1. *(10 points)* Check the assumptions that linear discriminant analysis makes and decide whether its use is justified.

   Hint: Appropriate plots can help to check the assumptions.

2. *(10 points)* Train a classifier using linear discriminant analysis. Remember to hold back a subset of the data for model evaluation.

   Hint: You can use implementations from Python packages that you can find in the `requirements.txt` file or implement lda yourself.

3. *(2 points)* Print out the accuracy of your model on the held back test set.

4. *(5 points)* Calculate the empirical model risk using $0-1$ loss on the same set as in subtask 3.

5. *(3 points)* Based on you answer in subtask 1, do you think the model comes close to the minimal risk achievable?

2 Image Classifier (img)

In this task you have to train an image classifier on data as exemplary shown in Figure 1. `X_train.csv` contains the data of flattened images of dimensions $28 \times 28 \times 1$, where a row corresponds to one flattened image. `y_train.csv` contains the corresponding labels. You are asked to solve the task within the formal requirements but are free to use any machine learning model you like. You are awarded up to 70 points for the task according to the description below.

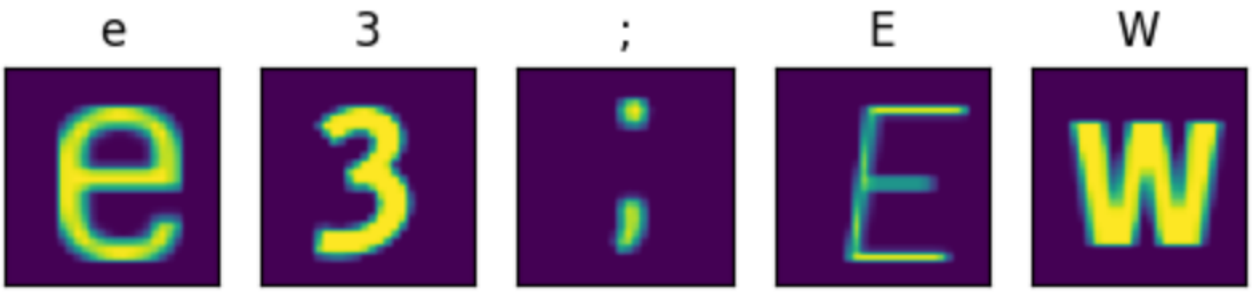

Figure 1: Example data.

2.1 Evaluation - Image Classifier

We will evaluate your model on a secret test set. You will be awarded points for your accuracy (**acc**; rounded to the next integer) according to the following scheme:

- **acc** $\geq 85\%$: 70 points
- $50\% <$ **acc** $< 85\%$: 2 points per percentage point above 50

If your accuracy is not equal to or does not exceed 85% (but is greater than 50%), you have the availability to 'earn back' half of the missing points for this task by:

- commenting and documenting your code well,
- using efficient implementations (e.g. use numpy functions, not for loops whenever possible), and
- using challenging methods beyond those covered in class.

Note: **acc** has to be greater than 50%. Otherwise we treat your model as not providing results (see below).

Examples:

- You train a well-performing model and achieve 80% accuracy. You can earn back up to 5 points and reach a maximum of 65 points for the task.
- If your model achieves 60% accuracy on the test data, you can earn back up to 25 points by submitting clean code and using interesting methods, i.e. you can reach at most 45 points.

What if your code does not run or provide any results? We will take limited time to try and make minor fixes, and will deduct some points from the final result, depending on how severe the problem was. However, if it is not quickly solvable, we will award you 30 points at best, depending on the quality of your code and the models you used. The latter also applies if your accuracy falls short of 50%.

Please pay close attention to the formal requirements! Good luck!