Foundations of Data Analysis - SS23
Lab assignment
Supervised learning

Due date: 09:45 am on 27.04.2023

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.

Note: In this assignment it is EXTREMELY IMPORTANT to follow the formal requirements outlined below. If you do not know how to use the provided templates, ask in the Moodle forum. Failing to comply with the requirements may result in your assignment receiving 0 points!

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 regression and lasso regression[1] (**lasso**). 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!

[1] see task 1 for an explanation

Formal Requirements

Download the following files from u:cloud.
Link: https://ucloud.univie.ac.at/index.php/s/uW0xNSY3PShZkOq ;
Password: T=nd7a2d2!n

- `lasso_data.csv` (task 1: **lasso**)
- `lasso_template.py` (task 1: **lasso**)
- `X_train.csv` (task 2: **img**)
- `y_train.csv` (task 2: **img**)
- `img_template.py` (task 2: **img**)
- `test_script.py` (task 2: **img**)
- `requirements.txt` (both tasks)

`lasso_data.csv` contains the synthetic data for the **lasso** task and `lasso_template.py` a corresponding template that you **can** use if you want to.

`X_train.csv` contains flattened images[2] of dimensions $44 \times 48 \times 3$, `y_train.csv` contains the corresponding labels (numbers originally corresponding to traffic signs). `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 further 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. For grading purposes we will use a script similar to `test_script.py`. Hence, you can use `test_script.py` to verify that your function works as intended.

We will execute your code in an environment created from the file `requirements.txt`[3]. 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[4]. 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

[2] of traffic signs
[3] In Python Version 3.9.12
[4] e.g. `conda create --name fda python=3.9.12` + installing packages

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 either `lasso` or `img` indicating the task. For task 1 (**lasso**), however, we also accept Jupyter notebooks. (Example: `luther_c_lasso.py`/`luther_c_lasso.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 template 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 Regression and Lasso (lasso)

In this task you are asked to perform linear regression and lasso (see below). You have to perform the steps on the data set `lasso_data.csv`. The data set consists of six independent variables $X_1, X_2, ..., X_6$ and a dependent variable Y. You are supposed to train a model for the regression target Y.
The data is sampled according to the following data generating process:

$$X_1 \sim \mathcal{N}(-5, 1), \quad X_2 \sim \mathcal{N}(10, 2), \quad X_3 \sim \mathcal{N}(20, 20),$$
$$X_4 = X_2, \quad X_5 \sim \mathcal{N}(1, 5), \quad X_6 \sim \mathcal{N}(3, 3),$$
$$\epsilon \sim \mathcal{N}(0, 1), \quad Y = 3 \cdot X_1 + X_2 + X_4 - 2 \cdot X_5 + \epsilon$$

Lasso Regression

In the lecture you have learned about linear regression and its analytic solution using empirical risk minimisation (ERM). In this exercise you will use a different method to choose a hypothesis h^* from the linear hypothesis class, namely regularised risk minimisation. Specifically, you are asked to perform lasso regression. Lasso introduces a regularisation term to the optimisation problem that penalises the size of the model parameters and hence may set them (closer) to zero. The strength of the penalty is governed by a parameter λ. The whole optimisation problem then looks like the following:

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - \mathbf{X}_i \beta)^2 + \lambda ||\beta||_1 \tag{1}$$

By penalising the absolute size of the model coefficient, lasso is able to perform feature selection (by setting coefficients to zero) and thus may lead to sparser models.

Note: We omit the intercept term for simplicity.

Hint: For model fitting, you *can* use functions that you can find in the packages provided and do not have to derive every step from scratch.

1. *(3 points)* Is it possible to solve the lasso optimisation problem analytically? *Explain.*
2. *(2 points)* Split the data into a train and a test set with appropriate test size.
3. *(5 points)* Fit a linear regression model for Y using *all* remaining variables. Use the training data for this task.
4. *(5 points)* Make a model prediction on unseen data and assess model performance using a suitable metric.
5. *(6 points)* Perform lasso regression using the same data as in subtask 3.

   Hint: The parameter λ sometimes is called α (or even differently). To choose a parameter, you can make different guesses or use cross validation.
6. *(2 points)* Compare model performance to the original linear model by using the same metric and test set as in subtask 4. What do you observe?
7. *(2 points)* Print out the model coefficients for both, the linear model and the lasso model.
8. *(5 points)* What do you observe comparing the estimated model coefficients? Was this result expected? *Explain.*

   Hint: If you do not manage to perform the model fitting, you can still find an answer to this question by looking at the data generating process and the introduction to lasso regression.

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 $44 \times 48 \times 3$, 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.

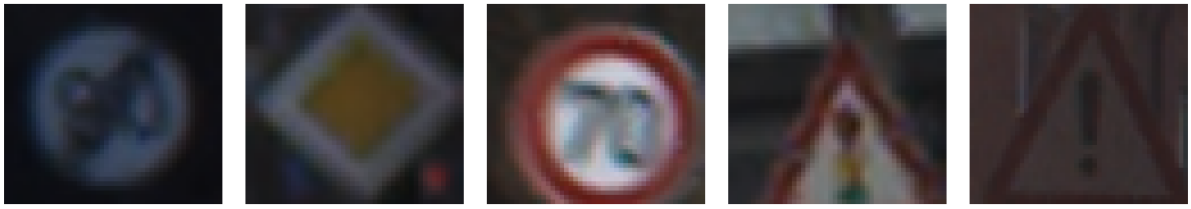

Figure 1: Example data.

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** $\geq 95\%$: 70 points
- $60\% <$ **acc** $< 95\%$: 2 points per percentage point above 60

If your accuracy is not equal to or does not exceed 95% (but is greater than 60%), 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 60%. Otherwise we treat your model as not providing results (see below).

Examples:

- You train a well-performing model and achieve 85% accuracy. For the model accuracy you get $(85 - 60) \cdot 2 = 50$ points. Additionally, you can earn back up to 10 points and reach a maximum of 60 points for the task.
- If your model achieves 70% accuracy on the test data, you get $(70 - 60) \cdot 2 = 20$ points for your accuracy. 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 60%. Note, that severe errors, e.g., failing to use the correct file type, will result in zero points.

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