Sarah Schröder, Alexander Schulz Bielefeld University

Introduction to Machine Learning (WS 2023/24)
2nd Project

Released: Tuesday, 05.12.2023.
Due: Please solve the exercises in groups of three and submit your report and your code as an executable python script (`.py`) to Moodle by **Tuesday, 19.12.2023, 11:59pm**.

- Please note that the course language is **English**. However, you might hand-in your report in German as well as in English, but be consistent. We will not substract any points for errors regarding language as long as your report is understandable.
- We provide a LaTeX template `report_template.tex` which might help you.
- There is a Q&A tutorial for this project on Monday, 11.12.2023. You can use this session to work on this sheet and/or ask your tutor if you get into trouble.
- Submit your *pdf* and your code as *py*-file (you can export a py-file out of jupyter-notebook) to the LernraumPlus!
- If you use any code from the internet put a link to the source as an comment into your code for reference.
- The project will be discussed during the tutorials on Monday, 08.01.2024.
- If you have any questions, please ask your tutor or write an email to intromachlearn@techfak.uni-bielefeld.de.

In this project you will work on both artificial and real world data. You can gain up to 90 points in this project.

Work on the tasks described below and write a **full-text** report on **max. seven** pages (the next pages will not be included in the grading). Additionally, an appendix containing the proof of task 1b and your figures is allowed. Your report should contain

- a description of the data sets used (now that you are partly working with real world data, your description might be a little more detailed here as you have information about what the features actually mean),
- descriptions of models, techniques and metrics used,
- a documentation of your experimental set-up (What choices did you make? Why?),
- your results and an analysis,
- plots (where necessary/useful) with proper legends/ descriptions,
- and, of course, a short introduction and conclusion.

When writing your report, please make sure that your descriptions are complete and precise. Based on your text we should be able to reproduce your pipeline and your results. Besides, you will have quite a lot of results. Consider one part of the task it to present them in a consist way. For example, you can consider visualizing your results in plots or creating tables. A good plot always contains axis labels, title and a legend or proper description.

When putting together your report pay attention to the structure. This is very important, as it is hard grasp work described in a badly structured report.

Please hand in your report as a PDF file and one python file containing all your source code. We will consider only your written report for the grading, perform a plagiarism check on your code and check whether it runs smoothly. To ensure this, please store the data in the same directory as your python file and do not use absolute paths.

Hint: When creating plots using matplotlib, you can save the current figure by `plt.savefig('path/to/file.eps', format='eps')`. If you are using LaTeX for the first time `www.tablesgenerator.com` might be a helpful resource for easily creating clean looking tables.

1 kNN Regression

(*16 Points*)

In the beginning of the lecture, we learnt that classification is only one technique in supervised learning. Another one is regression. Come up with a method to adapt the kNN classifier for regression:

(a) Describe your strategy to adapt the kNN classifier from classification to regression

(b) Implement your proposed method and cross-validation (own coding required for both!)

(c) Evaluate your models performance on dataset4 and compare to a dummy regressor[1] (use cross validation)

(d) Test how your model behaves for different values of $k \in [2, 100]$. Report the model's performance for three values of k where you observe (i) good generalization, (ii) overfitting, (iii) underfitting. How can you tell which is which? It may be interesting to plot the predictions.

2 Artificial Data

(*22 Points*)

We will first consider an artificial data set.

(a) Train three different regression models: A linear, a polynomial regression model and kNN Regression[2]. Evaluate them with cross-validation using R2-score on data set 5.

(b) Apply standardization to data set 5 and rerun all experiments.

(c) (own coding required!) Implement a function for computing and plotting the so-called learning curve (see description below). Compare the learning curves of the different models. Compute the learning curve with R2 for 1 to n samples for linear regression and for k to n for kNN regression. In this task, a simple train test split is sufficient, you do not need to do cross validation here.

Note: In utils.py you can find a function `plot_2d_regression` to plot 2 dimensional regression data.

Learning Curve: Learning curves show how well the model performs when trained with a growing amount of data samples. First of all, it is necessary to split the data set into a train and a test set. In order to compute the learning curve, we repeatedly train a model with a growing number of training samples (from the train set) starting by one and stopping with the number of training samples. We evaluate each of these models on the part of the training set which was already used for training, and on the entire test set. Finally, we plot the results over the number of samples used for training.

3 Real world data

(*22 Points*)

Now, we move on to real world data. More precisely, a data set about house prices is given. Load the data `real_world.npz` with the field names `X`, `y` and `features`. `X` gives you the data matrix, `y` the corresponding outputs. You can find the feature names in `features`.

(a) As for the artificial data, train Linear and kNN Regression and evaluate them with the R2-score using cross-validation.

(b) Compare and analyze the results. Additionally consider the learning curves of your models, describe and analyze them. You do not need to use cross validation. Hint: here it might make sense to not add one sample after another but to compute the learning curve in bigger step sizes - for example one of 10.

(c) In this task we consider additional data from `new_data.npz` (same fields `X`, `y`, `features`), where some values are missing. Implement two approaches to impute missing values: (1) Set the missing value as the feature mean, (2) Derive the missing value from the neighbor samples. Train Linear and kNN Regression on `real_world.npz` and evaluate their performance on the new data with each approach. Explain which approach works better and why.

[1] The dummy regressor could simply guess the values by outputting the mean of training values or doing a simple linear regression.

[2] You may use a pipeline with polynomial preprocessing as in Project 1. If you want to use kNN Regression, but had trouble with task 1, you may use the sklearn implementation.

4 Report Quality

(*30 Points*)

In addition to the implementation and analysis of your results, we will score the completeness and overall quality of the report. Consider the bullet points from the first page.