UCL DEPARTMANT OF MEDICAL PHYSICS AND BIOMEDICAL ENGINEERING

Undergraduate / Postgraduate Assessed Coursework Tracking Sheet

Module Code:	MPHY0041
Module Title :	Machine Learning in Medical Imaging
Coursework Title :	Assessed Coursework
Lecturer:	Dr. Andre Altmann
Date Handed out:	Thursday, November 2nd 2023
Student ID (Not Name)	

Submission Instruction: Before the submission deadline, you should digitally submit your source code and generated figures (a single jupyter notebook file including your written answers). In case you submit multiple files, all files need to be combined in one single zip file and submitted on the module page at UCL Moodle.

Coursework Deadline:	Thursday, **December 7th 2023 at 16:00** at UCL Moodle submission section
Date Received	
Date Returned to Student:	

The Department of Medical Physics and Biomedical Engineering follows the UCL Academic Manual with regards to plagiarism and coursework late submission.
UCL Policy on Plagiarism
UCL Policy on Late Submission of Coursework

If you are unable to submit on-time due to extenuating circumstances (EC), please refer to the UCL Policy on Extenuating Circumstances and contact our EC Secretary at medphys.teaching@ucl.ac.uk as soon as possible.
UCL Policy on Extenuating Circumstances

Please indicate what areas of your coursework you particularly would like feedback on:

Mark (%):

Please note that the mark is provisional and could be changed when the exam boards meet to moderate marks.

Please note: Please submit a single jupyter notebook file for Exercises 1, 2, 3 and 4. The file should contain code, plots and comments that help the understanding of your answers. You can give your written answers as a Markdown within the jupyter notebook. **Do not use any AI-support to solve these exercises 1, 2, and 3 (i.e., no chatGPT etc.), using tools such as co-pilot for exercise 4 is permitted.**

The provided jupyter notebook `Notebook_MPHY0041_2324_CW1.ipynb` contains the individual gap codes/functions for Exercise 2 and the functions provided for Exercise 4. You can use this notebook as basis for your submission.

1. Load the dataset '`PPMI_DATSCAN.csv`' it contains data from Dopamine Transporter Scan (DaT scan) of two brain regions in the left and right hemisphere of the brain: caudate and putamen. DaT scan is a single-photon emission computed tomography (SPECT) method to measure the loss of dopaminergic neurons in diseases such as Parkinson's disease. It also contains the values of a test for motor abilities (MDS UPDRS Part II). The dataset comprises Healthy Controls (HC), Parkinson's disease (PD) and Scans Without Evidence of Dopaminergic Deficit (SWEDD). The column 'COHORT_DEFINITION' denotes the diagnosis.
    a) Remove SWEDD subjects from the dataset. Compute means for DaT scan in the right putamen (DATSCAN_PUTAMEN_R) for the 'HC' (μ_{HC}) and the 'Parkinsons' (μ_{PD}) groups. In addition, compute the standard deviation (σ) for DaT scan in the right putamen in that dataset. Assume that the data follow a Gaussian distribution:
    $$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2},$$
    with the means and standard deviations as computed above. Compute the decision boundary between the two disease groups (with the prior probabilities $\pi_{\mathrm{HC}} = \pi_{\mathrm{PD}} = 0.5$). [5]

    b) Using `sklearn` functions, train a `LinearRegression` to separate HC from PD subjects using DaT scan values for the right putamen and MDS UPDRS Part II score (NP2PTOT) as inputs. Generate a scatter plot for DaT scan in the right putamen and NP2PTOT using different colours for two diagnostic groups. Compute the decision boundary based on the linear regression and add it to the plot. [4]

    c) The previous analyses ignored the subjects with SWEDD. Going back to the full dataset, compute means for all three groups for DaT scan in the right putamen and the NP2PTOT score as well as the variance-covariance matrix Σ. Use these to compute linear decision boundaries between all pairs of classes (with the prior probabilities $\pi_{\mathrm{CN}} = \pi_{\mathrm{MCI}} = \pi_{\mathrm{Dementia}} = 0.33$). Generate a new scatterplot and add the three decision boundaries. [7]

2. Here we complete implementations for different algorithms using different loss function with gradient descent.
    a) Implement the algorithm to fit linear regression with Residual Sum of Squares (RSS) as the loss function: $L\big(y, f(x)\big) = \big(y - f(x)\big)^2$. The loss for the dataset should be $L(\beta) = \frac{1}{2N}\sum_{i=1}^{N} L\left(y_i, f_\beta(x_i)\right)$. Furthermore, use L_2 regularization of the β coefficients: $\lambda * \sum_{j=1}^{p} \beta_j^2$. The function `fit_RSSl2_GRAD` contains a few gaps that need to be filled for the function to work. Load the dataset '`sim_data.csv`' and use your completed function to train the model (y is the outcome; $x1, x2, x3, x4$ are the inputs). Build 3 models with different values for λ: $0.0, 0.4, 0.1$. Provide the coefficients, make predictions for the data in '`sim_test.csv`', plot $x1$ against the observed values (y), and the predicted values by all three models ($\hat{y}$). [4]

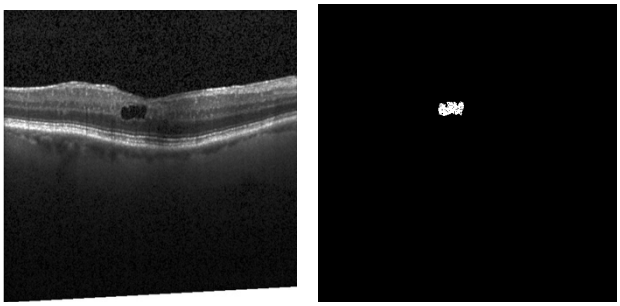
    b) Implement the algorithm to fit linear regression with `LogCosh` as the loss function: $L\big(y, f(x)\big) = \log\big(\cosh\big(y - f(x)\big)\big)$. The function `fit_logcosh_GRAD` contains a few gaps that need to be filled for the function to work. Load the dataset '`sim_data.csv`' and use your completed function to train the model (y is the outcome; $x1, x2, x3, x4$ are the inputs). Provide the coefficients, make predictions for the data in '`sim_test.csv`', plot $x1$ against the observed values and the predicted values ($\hat{y}$). [4]

    c) Build a more flexible model that adapts to *noise* in the data. We assume that the target variable follows a normal distribution $y_i \sim N(\mu(x_i), \sigma(x_i))$, where both parameters (mean and standard deviation) are linear functions in x. That is: $\mu(x) = \beta_0 + \vec{\beta}x$ and $\sigma(x) = \theta_0 + \vec{\theta}x$. The model parameters can be fitted using maximum (log-)likelihood estimation. Define the loss function over the dataset with N data points. Derive its partial derivatives with respect to the model parameters $\beta_0, \vec{\beta}, \theta_0, \vec{\theta}$. Implement the optimization using gradient descent. Load the dataset '`sim_data.csv`' and use your completed function to train the model (y is the outcome; $x1, x2, x3, x4$ are the inputs). Provide the coefficients for $\mu(x)$ and $\sigma(x)$ make predictions for the data in '`sim_test.csv`', plot $x1$ against the observed values (y) and the predicted values for mean and standard deviation. Explain why the model does not work as intended. [7]

3. Researcher A (RA) is working on a machine learning task. RA aims to classify lung CT scans into people with chronic obstructive pulmonary disease (COPD) and healthy controls. The dataset comprises 72 people with COPD and 94 controls, resulting in 154 COPD images and 207 control images. The CT data has the dimension of 70x70x35 voxels (each of size $3mm^3$). Because of the high dimensionality of the data, RA decides to use Support Vector Machines with a linear kernel and selects the cost (C) parameter to be 10.0. RA runs a 5-fold cross-validation and measures the correlation between the output and the actual labels. The machine learning model achieved an accuracy of 0.82 and RA is now convinced the model is ready to be used in the clinic.

   State four corrections to the analysis proposed by RA and provide your reason for making each correction. [8]

4. This exercise uses retinal optical coherence tomography (OCT) images. The task to be solved is to automatically segment Intraretinal Cystoid Fluid (ICF) in these images. The input images are RGB images with 512x512 pixels (below left) and the output should be a binary matrix of size 512x512, where a 1 indicates the presence of fluid (below right).



   The `icf.zip` archive contains three sets of images: training, validation, test. For training, there are 400 OCT images paired with their ground truth (i.e., masks). For instance, `train/10DME_F/images/10DME_F.jpeg` is the OCT image and `train/10DME_F/masks/10MASK_DME_F.png` is the corresponding ground truth. Use the function provided in `create_training_set` to randomly sample 500 patches of size 11x11 from the 400 training images to generate an initial dataset. Of note, the resulting dataset is heavily imbalanced (i.e., mostly '0' labels). Use the function provided in `sub_sample` to create the training dataset with 10,000 samples where '1' is oversampled with a 200:1 ratio.

   a) Using `sklearn`, train an SVC model to segment the fluid. Optimize kernel choice (e.g., RBF or polynomial with degree 3) and the cost parameter (C in the range 0.01 to 100) using cross-validation. Measure performance using the Area Under the ROC Curve (roc_auc) and plot the performance of the kernels depending on the C parameter. (Hint: when SVC seems to take an endless time to train, then change your choice of C parameters; large C parameters → little regularization → long training time.) [6]

b) Based on your result from a) select the best model parameters and make predictions of the 50 images in the validation dataset. Compute the DICE coefficient and roc_auc for each image. Display the original image, the ground truth, and your segmentations for any 5 images in your validation set. Provide the average DICE coefficient and roc_auc for the entire validation dataset. [6]

c) Instead of the SVC, train a tree-based ensemble classifier and make predictions for the validation images. Report the average DICE coefficient for the entire validation set. What performs better the SVC or the tree ensemble? [3]

d) Use the tree-based ensemble method and explore how the amount of training data (i.e., sub sample size: 500, 1000, 5000, 10000, 20000), the patch dimensions (3x3, 7x7, 11x11, 15x15, 19x19), and the sampling ratio (50,100,200,400,800) affects the performance on the validation set. [5]

e) Modify the function `preprocess_img` to add at least one additional channel (feature) that will improve the performance on the validation data. (Hint: the `scikit-image` library offers various filters that might provide good features.) Report how these features influence the model's performance. [4]

f) Using your best combination of training data size, patch dimension, sampling ratio (from d) and filter set (from e), estimate the performance on unseen samples from the test set. Provide average DICE coefficient for the entire test set. [2]