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:	Friday, February 3rd 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:	Friday, **March 3rd 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.

The provided jupyter notebook `Notebook_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 '`AD_CW1.csv`' it contains data from whole brain amyloid PET as well as the volume of the hippocampus normalized to head size. Amyloid PET is used to quantify the amount of the amyloid protein in the brain. This protein is elevated in people with Alzheimer's disease. Hippocampal volume is simply the size of the hippocampus, a brain region responsible for storing memories. During Alzheimer's disease the hippocampus degrades heavily. The dataset comprises Cognitively Normal (CN) participants as well as people with Alzheimer's disease (Dementia) and Mild Cognitive Impairment (MCI). The column *'DX'* denotes the diagnosis. We are interested in the decision boundaries for different groups. For the derivation of decision boundaries, please show your reasoning/workings in addition to the final formula for the boundary.
    a) Remove MCI subjects from the dataset. Compute means for amyloid PET (*Amyloid*) for the 'CN' (μ_{CN}) and the 'Dementia' (μ_{AD}) groups. In addition, compute the shared standard deviation (σ) for Amyloid in CN and AD. 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 $\pi_{\mathrm{CN}} = \pi_{\mathrm{AD}} = 0.5$). [5]

    b) Using `sklearn` functions, train a `LinearRegression` to separate CN from Dementia subjects using both Amyloid and hippocampal volume (*HV_ratio*). Generate a scatter plot for amyloid and HV_ratio using different colours for the 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 MCI. Going back to the full dataset, compute means for all three groups for Amyloid and HV_ratio as well as the variance-covariance matrix Σ (Hint: Use vector representations, the means have dimension 2x1 and Σ has dimension 2x2). 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 classification algorithms.
   a) Implement the algorithm to fit Rosenblatt's Perceptron (Week 5 lectures). The function `fit_rosen_perceptron` 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. Run the model five times and provide the estimated coefficients, plot the data and the decision boundary. [5]

   b) The function `fit_LogReg_GRAD` aims to implement Logistic Regression using gradient descent (Week 3 lectures). However, there are still a few gaps in the code. Complete the computation of the cost (`J(β)`) as well as the gradient step, i.e., the update of the beta coefficients (β). (Hint: Watch the signs. Gradient descent aims to *minimise* the cost; however, Logistic Regression is fitted by *maximising* the log likelihood). Use your function to train a model that separates the two classes in the '`sim_data_nonsep.csv`' dataset. Provide the estimated beta coefficients, plot the data and the decision boundary. [5]

   c) The function `fit_HingeL2_GRAD` aims to implement a linear classifier using Hinge Loss (defined as: $\max(0, 1 - y * \hat{y})$) and L2 regularization. The full objective function is:

$$J(\beta) = \sum_{i=1}^{N} \max\left(0, 1 - y_i * f_{\beta_0,\beta}(x_i)\right) + \lambda \frac{1}{2} ||\beta||^2$$

   Where

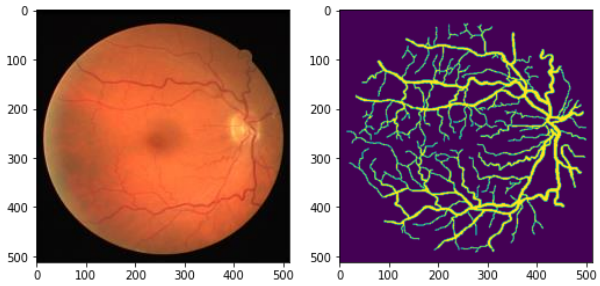
$$f_{\beta_0,\beta}(x) = \beta_0 + \beta^T x$$

   implements a linear function with intercept β_0 and a weight vector β. Complete the gaps in gradient descent code. Use the completed function to train a model on the data in '`sim_data_nonsep.csv`' with regularization parameters $\lambda =$ 0.01, 1, 100. Provide the resulting beta coefficients. Plot the data and the decision boundary. [5]

3. Researcher A (RA) is working on a machine learning task. RA aims to classify brain MRI scans into people with Alzheimer's disease (AD) and Cognitively Normal (CN) controls. The dataset comprises 63 people with AD and 104 CNs. The MRI data has the dimension of 100x100x100 voxels (each of size $2mm^3$). The imaging data has been pre-processed (i.e., spatially aligned by co-registering to a template). Because of the high dimensionality of the data, RA decides to use Ridge Regression and sets the regularization parameter to 1.0 to train the model on the entire dataset. After training RA applies the trained model to the same dataset and measures the correlation between the output and the actual labels. The machine learning model achieved a correlation of 0.3 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 fundus images. The task to be solved is to automatically segment the blood vessels 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 a vessel (below right).



The `fundus.zip` archive contains three sets of images: training, validation, test. For training, there are 80 fundus images paired with their ground truth (i.e., masks). For instance, `train/images/0.png` is the fundus image and `train/masks/0.png` is the corresponding ground truth. Use the function provided code in `create_training_set` to randomly sample 50 patches of size 7x7 from the training images to generate a training dataset.

a) Using `sklearn`, train an SVC model to segment vessels. Optimize kernel choice (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 both 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 10 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 the 10 images and provide the average DICE coefficient and roc_auc for the dataset. [4]

c) Instead of the SVC, train a tree-based ensemble classifier and make predictions for the validation images. Report the average DICE coefficient. What performs better the SVC or the tree ensemble? [5]

d) Use the tree-based ensemble method and explore how the amount of training data (i.e., patches sampled: 50, 100, 500, 1000) and the patch dimensions (5x5, 7x7, 9x9, 11x11) affects the performance on the validation set. [5]

e) Modify the function `preprocess_img` to extract at least two more features from the fundus images that can be used by the classifier. (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, and filter set, estimate the performance on unseen samples. Provide average DICE coefficient. [2]