ML4ENG Coursework – Part 1

Deadline for submission (enforced by Keats): March 7 2022 at 16:00.

1 GENERAL GUIDELINES

This coursework deals with binary classification of breast cancer using Wisconsin data [1]. To start, download the data set `dataset_wisconsin.mat` and the template file `cw_template.m` from the module's Keats website. Once this is done:

1. Change the `cw_template.m` to your k number. In the following, we will refer to this file as `k12345678.m`.
2. Open the `k12345678.m` file with your MATLAB editor. Note that the file contains a preamble, referred to as main body, which you should **not** modify, and the definition of several functions.
3. Follow the Instructions (Section 3 of this document) to fill in the details of the functions in the template file. The main body of k12345678.m has been divided into sections, with each section containing one or more functions to be completed. The functions in the k12345678.m file have been numbered according to the numbered list below in Section 3 (Instructions).
4. Once you have written the functions, verify `k12345678.m` runs without errors when the file is included in a folder containing **only** the file itself and the data set `dataset_wisconsin.mat`.
5. Check that **no** MATLAB toolbox was used (you can type in the command window `license('inuse')` and verify only standard `matlab` functions are used).
6. Submit only the `k12345678.m` file on Keats. No other files are allowed.

2 DATA SET

The file `dataset_wisconsin.mat` contains a data set $\mathcal{D} = \{x_i, t_i\}_{i=1}^{N}$, which consists of $N = 569$ examples. Each example consists of:

1. Input vector x_i in $\mathbb{R}^{30}$, encompassing $d = 30$ features extracted from a contour drawn over an image and its enclosed grey levels pixels.
2. Its corresponding binary label $t_i \in \{0,1\}$, where 1 stands for malignant and 0 for benign, as diagnosed by an expert physician.

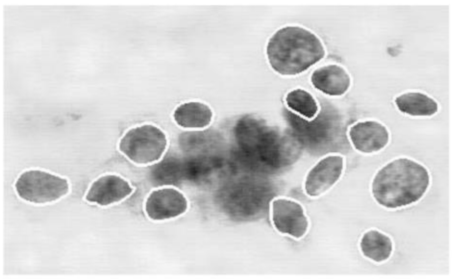

Figure 2: Snakes After Convergence to Cell Nucleus Boundaries
These contours are the final representation of the cell nuclei boundaries after the user is satisfied with the convergence of the snakes. This interactive process takes about two to five minutes.

Figure 1: The contours made by the examiner, which are transformed into 30 features (mean, standard deviation and worst values of 10 base features). Figure taken from [3].

The data is loaded into the workspace to have

Name	**Size**	**Type**	**Description**
`t`	$N \times 1$	Logical	Diagnosis (binary label): 1 = malignant and 0 = benign.
`X`	$N \times d$	Double	Data matrix (samples vectors as rows)
`x_titles`	$1 \times d$	String	Description for the *d* features in x

The input sample vector is denoted as

$$x = [x^{(1)} \quad \dots \quad x^{(d)}]^T,$$

and the inputs of the data sets are given by stacking up N samples

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & \cdots & x_1^{(d)} \\ \vdots & \ddots & \vdots \\ x_N^{(1)} & \cdots & x_N^{(d)} \end{bmatrix}.$$

The labels are also stacked up, forming the vector

$$t = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix}.$$

The entries of the vector `x_titles` annotate the d features. Specifically, in the first part we focus on the first two features: $x^{(1)}$ for `radius_mean` and $x^{(2)}$ for `texture_mean`.

3 Instructions for Completing the Coursework

Section 1

In this section, we split the full data set $\mathcal{D}$ into training $\mathcal{D}^{tr}$ and test set $\mathcal{D}^{te}$ using 70-30 ratios.

1. [10 points] Design the function

   `function [X_tr, t_tr, X_te, t_te]= split_tr_te(X, t, eta)`

   that splits the input data {`X`,`t`} set into two disjoint data set. The training set {`X_tr`,`t_tr`} should have the **first** $N^{tr} = \mathrm{round}(\eta N)$ samples and labels, and the test set {`X_te`,`t_te`} the remaining $N^{te} = N - N^{tr}$ **last**. Here η stands for the ratio of the training data set size from the entire data set.

Section 2

We denote the following two single-feature-based thresholding hard predictors

$$\hat{t}_1(x) = \begin{cases} 1, & x^{(1)} > 14.0 \\ 0, & \text{otherwise} \end{cases}$$

$$\hat{t}_2(x) = \begin{cases} 1, & x^{(2)} > 20.0 \\ 0, & \text{otherwise} \end{cases}$$

2. [5 points] Design the function

   `function t_hat1 = hard_predictor1(X)`

   that produces the column vector $\hat{t}_1(X) = [\hat{t}_1(x_1) \quad \dots \quad \hat{t}_1(x_N)]^T$ given arbitrary input samples.

3. [5 points] Design the function

   `function t_hat2 = hard_predictor2(X)`

   that produces the column vector $\hat{t}_2(X) = [\hat{t}_2(x_1) \quad \dots \quad \hat{t}_2(x_N)]^T$.

Each of these functions runs separately over the training and test sets.

Section 3

In this section, we calculate the sensitivity and specificity of these two predictors. For some predictor $\hat{t}$, they are defined as

$$\text{sensitivity} = \mathbb{P}\mathrm{r}(\hat{t} = 1 | t = 1),$$

$$\text{specificity} = \mathbb{P}\mathrm{r}(\hat{t} = 0 | t = 0).$$

Since the population distribution is unknown, we will use the empirical test distribution $p_{\mathcal{D}^{te}}(x, t)$ as an estimation for the probability.

4. [10 points] Design the function

   `function [sens, spec] =`
   `sensitivity_and_specificity``(t_hat, t)`

   that computes the empirical sensitivity and specificity using the prediction vector and the true label target vector, both with the same length. Note this function is called twice at the main body with different predictors.

Section 4

Here, we estimate the detection error loss of both predictors over the test set.

5. [5 points] Design the function

   `function loss =` **`detection_error_loss`**`(t_hat, t)`

   that computes the empirical detection error loss of binary predictions `t_hat` with respect to the true targets `t`.

Section 5

6. [10 points] Add up to two lines of text in `function` **`discussionA`**`()` to discuss which of the two predictors $\hat{t}_1$ and $\hat{t}_2$ is better to predict breast cancer on its own, and why. Support using relevant metrics.

Section 6

In this section, instead of fixing a predictor, we train it based on the available training data. To this end, we consider two linear predictors using different features. Their model class consist of predictors of the form

$$\hat{t}_3(x|\theta_3) = \theta_3^T \cdot u_3(x),$$

$$\hat{t}_4(x|\theta_4) = \theta_4^T \cdot u_4(x).$$

where the first feature vector uses the mapping

$$u_3(x) = \left[1, x^{(1)}, x^{(2)}\right]^T,$$

whereas the second uses also second order terms of the first two features

$$u_4(x) = \left[1, x^{(1)}, x^{(2)}, \left(x^{(1)}\right)^2, \left(x^{(2)}\right)^2, x^{(1)}x^{(2)}\right]^T.$$

Recall x is the d-dimensional input feature vector. To train the predictors, we optimize the model parameter vectors $\theta_3 \in \mathbb{R}^3$ and $\theta_4 \in \mathbb{R}^6$ using the quadratic loss by solving a standard least squares problem over the two new training data matrices. The LS function is provided.

7. [5 points] Design the function

   `function out = X3(X)`

   that produces the data matrix of size $N \times 3$ using the feature mapping $u_3(\cdot)$

$$X_3 = \begin{bmatrix} (u_3(x_1))^T \\ \vdots \\ (u_3(x_N))^T \end{bmatrix}.$$

8. [5 points] Design the function

   `function out = X4(X)`

   that produces the data matrix of size $N \times 6$ using the feature mapping $u_4(\cdot)$

$$X_4 = \begin{bmatrix} (u_4(x_1))^T \\ \vdots \\ (u_4(x_N))^T \end{bmatrix}.$$

Section 7

This section visualises $\hat{t}_3$ and $\hat{t}_4$ on the two-dimensional space of $x^{(1)}$ and $x^{(2)}$. For that, it spans the space using a grid and predicts for each sample in that grid `X_gr` the outcome of the two predictors. Since the LS prediction $\theta^T \cdot u(x)$ is continuous and not binary, proper clipping to $[0,1]$ is done, and hard thresholding as the decision border is shown

$$\hat{t}^{hard}(x|\theta) = \begin{cases} 1, & \theta^T \cdot u(x) > 0.5 \\ 0, & \text{otherwise} \end{cases}.$$

The labelled test set is illustrated on top of the predictors' outcomes.

9. [5 points] Design the function

   `function out = linear_combiner(X, theta)`

   that applies the predictor $\theta^T \cdot u(x)$ to each input features sample in data matrix `X` (i.e., to each row $u(x)$ of the matrix) using `theta` for θ. The output is a real-valued vector whose number of elements equals the number of samples in `X`.

Section 8

We further evaluate the mean square error loss, and the detection error loss on its binary targets of the two predictors $\hat{t}_3$ and $\hat{t}_4$. We use the test set for that.

10. [5 points] Design the function

    `function out = mse_loss(t_hat, t)`

    that computes the empirical mean square error loss of prediction `t_hat` using the true labels `t`, both vectors of the same length.

Section 9

11. [10 points] Add a two-line comments in `function` **`discussionB`**`()` about which predictor has higher model capacity, and whether the higher complexity was useful for this problem. Use the calculated metrics to support your conclusion.

Section 10

In this section, we calculate the test loss using the entire input space by using the features

$$u_5(x) = [1, x^T]^T = \left[1, x^{(1)}, \dots, x^{(30)}\right]^T$$

12. [5 points] Design the function

    `function out = X5(X)`

    that produces the data matrix of size $N \times 31$ using the feature mapping $u_5(\cdot)$

$$X_5 = \begin{bmatrix} (u_5(x_1))^T \\ \vdots \\ (u_5(x_N))^T \end{bmatrix}.$$

Section 12

In this section, we quantify how the full-features least square solution test loss is sensitive to the amount of training data samples. For this purpose, we run over a fraction of 10:3:100 percent of the full training data set, and plot a graph of test loss vs. the ratio of the training set used for least squares .

13. [20 points] Design the function

    `function v_loss_LS =`
    `loss_vs_training_size``(X_tr, t_tr, X_te, t_te, v_ratio_tr)`

    that outputs a vector of the same size of `v_ratio_tr`, whose elements include the detection error test loss (of the entire $\mathcal{D}^{te}$) of the full feature LS using the **first** `round(v_ratio_tr(i)*`N^{tr}`)` input samples and targets of the training set. You may use auxiliary functions already implemented in `k12345678.m`.

4 REFERENCES

[1] https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

[2] https://uk.mathworks.com/academia/tah-portal/kings-college-london-30860095.html

[3] W. Nick Street, W. H. Wolberg, and O. L. Mangasarian "*Nuclear feature extraction for breast tumor diagnosis*", Proc. SPIE 1905, Biomedical Image Processing and Biomedical Visualization, (29 July 1993); https://doi.org/10.1117/12.148698