Experiment 4

Breast Cancer Diagnosis via Logistic Regression

1. Objective

The goal of this experiment is to develop a computer program for automatic diagnosis of breast cancer based on logistic regression where the logistic loss function is defined by a dataset provided by Dr. Wolberg from General Surgery Department, University of Wisconsin, Madison, WI in 1990's. The dataset contains 30 carefully selected features from each of 569 patients [R1],[R2]. The same dataset has also been made available from the UCI Machine Learning Repository [R3]. The parameters in the logistic regression model are optimized by minimizing the logistic loss function mentioned above. In this experiment, the optimization is performed using the gradient descent (GD) algorithm, see Sec. 2.3 of the course notes.

References

[R1] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, "Nuclear feature extraction for breast tumor diagnosis," in *IS?T/SPIE Int. Symp. Electronic Imaging: Science and Technology*, vol. 1905, pp. 861-870, San Jose, CA., 1993.

[R2] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," AAAI Tech. Report SS-94-01, 1994.

[R3] UCI Machine Learning, http://archive.ics.uci.edu/ml, University of California Irvine, School of Information and Computer Science.

2. Background and Data Pre-Processing

2.1 Background

As described in the literature [R1][R2], early detection of breast cancer is enhanced and unnecessary surgery avoided by diagnosing breast masses from Fine Needle Aspirates (FNA's). A graphical interface has been developed to compute nuclear features interactively. In order to obtain objective and precise measurements, a small region of each breast FNA is digitized, resulting in a 640×400, 8-bit-per-pixel gray scale image. An image analysis program uses a curve-fitting program to determine the boundaries of nuclei from initial dots placed near these boundaries by a mouse. A portion of one such processed image is shown in Figure E4.1. Ten features are computed for each nucleus that include area, radius, perimeter, symmetry, number and size of concavities, fractal dimension (of the boundary), compactness, smoothness (local variation of radial segments), and texture (variance of gray levels inside the boundary). The mean value, extreme value, and standard error of each of these cellular features are computed, resulting in a total of 30 real-valued features for each image. All feature values are recorded with four significant digits.

The dataset to be used in this experiment collects these 30 feature values from each of 569 patients of which 357 patients were diagnosed as benign while the other 212 patients were diagnosed as malignant. In what follows, the dataset will be referred to as WDBC which stands for Wisconsin Diagnostic Breast Cancer.

Using this dataset, the objective of this lab experiment is to develop a classifier to classify a new and unused sample either to benign or to malignant.

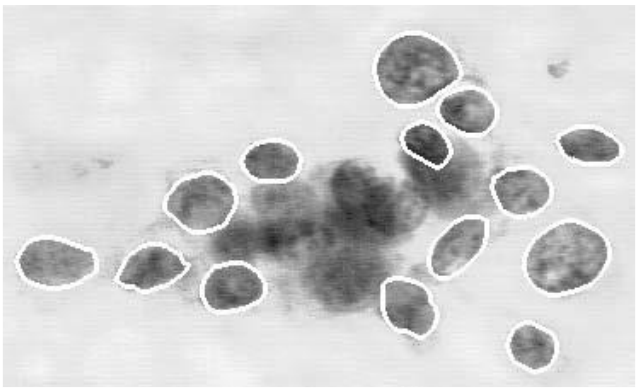

Figure E4.1 (from [R2]) A sample image where the cell nucleus boundaries are identified using an active contour model known as a "snake".

2.2 Pre-processing the data

Before a useful loss function can be defined, it is often necessary to pre-process the original dataset acquired from raw measurements. In real-world problems, numerical ranges of different features can be vastly different because different features represent different physical or artificial characteristics of the data. It turns out that a loss function defined by a dataset whose features vary in widely different numerical ranges is often more difficult to handle relative to that defined in a domain with a normalized scale for all features. In the case of WDBC, its 1^{st}, 3^{rd}, and 4^{th} features represent radius (which is the mean value of distances from center to points on the perimeter), perimeter, and area of a cell nuclei, respectively. The value of a radius is typically in the range 20, while the perimeter and area are in the vicinity of 120 and $1{,}200$, respectively. Under the circumstances, the data in WDBC need to be normalized first.

Below we describe a simple method to normalize a dataset $\mathcal{D} = \{(\boldsymbol{x}_n, y_n), n = 1, 2, ..., N\}$ where $\boldsymbol{x}_n \in R^{d\times 1}$ and label y_n is set to $y_n = -1$ if $\boldsymbol{x}_n$ is associated with a patient diagnosed as benign, and y_n is set to $y_n = 1$ otherwise.

Step 1 Form data matrix $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N \end{bmatrix}$ of size d by N and denote its ith row by $\boldsymbol{z}_i$, namely,

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \vdots \\ \boldsymbol{z}_d \end{bmatrix}$$

Note that each vector $\boldsymbol{z}_i$ contains N components which are the ith features of the N data samples in $\mathcal{D}$.

Step 2 Write vector $\boldsymbol{z}_i$ as $\boldsymbol{z}_i = \begin{bmatrix} z_1^{(i)} & z_2^{(i)} & \cdots & z_N^{(i)} \end{bmatrix}$ and compute its mean and variance as

$$m_i = \frac{1}{N}\sum_{k=1}^{N} z_k^{(i)}, \quad \sigma_i^2 = \frac{1}{N-1}\sum_{k=1}^{N}(z_k^{(i)} - m_i)^2$$

This step is carried out for $i = 1, 2, \ldots, d$.

Step 3 Normalize $\boldsymbol{z}_i$ as

$$\hat{\boldsymbol{z}}_i = \frac{1}{\sigma_i}\begin{bmatrix} z_1^{(i)} - m_i & z_2^{(i)} - m_i & \cdots & z_N^{(i)} - m_i \end{bmatrix}$$

so that the normalized $\hat{\boldsymbol{z}}_i$ has zero mean and unity variance.

Step 4 Finally, the normalized dataset is constructed as

$$\hat{\boldsymbol{X}} = \begin{bmatrix} \hat{\boldsymbol{z}}_1 \\ \hat{\boldsymbol{z}}_2 \\ \vdots \\ \hat{\boldsymbol{z}}_d \end{bmatrix} \tag{E4.1}$$

3. Solving Logistic Regression Problem for Breast Cancer Diagnosis

3.1 Preparing the data

WDBC is available from the course website as two separate data matrices:

• **`D_bc_tr.mat`:** This matrix will be used for training purposes. It is of size 31×480 whose first 30 rows contain 30 medical features for each of 480 patients. The corresponding labels are contained in the 31th row of **`D_bc_tr`**. Label "-1" is assigned to the samples associated with those diagnosed as benign, while label "1" is assigned to the samples associated with those diagnosed as malignant.

• **`D_bc_te.mat`:** This matrix will be used for test purposes. It is of size 31×89 whose first 30 rows constitute 30 medical features for each of 89 patients. The corresponding labels are contained in the 31th row of **`D_bc_tr`**.

Once the above data sets have been down-loaded, the train and test data are normalized *separately* as follows (see Sec. 2.2 above).

```
Xtrain = zeros(30,480);
for i = 1:30
    xi = D_bc_tr(i,:);
    mi = mean(xi);
    vi = sqrt(var(xi));
    Xtrain(i,:) = (xi - mi)/vi;
end
Xtest = zeros(30,89);
for i = 1:30
    xi = D_bc_te(i,:);
    mi = mean(xi);
    vi = sqrt(var(xi));
    Xtest(i,:) = (xi - mi)/vi;
end
```

and the labels associated with the train and test data are produced using

```
ytrain = D_bc_tr(31,:);
ytest = D_bc_te(31,:);
```

Summarizing, above code generates two data sets:
Xtrain is a matrix of size 30 by 480 containing features from 480 patients as train data;
ytrain contains 480 labels associated with the train data;
Xtest is a matrix of size 30 by 89 containing features from 89 patients as test data;
ytest contains 89 labels associated with the test data.

3.2 Logistic Regression

Given a dataset $\mathcal{D}$, logistic regression studied in Sec. 2.1 of the course notes is applicable to two-category classification problems. In what follows, it is assumed that dataset has been normalized, and for notation simplicity it is still denoted by $\mathcal{D} = \{(\boldsymbol{x}_n, y_n) \text{ for } n = 1, 2, ..., N\}$ with $N = 480$. The classification is performed in two steps.

(i) Minimize the objective function

$$f(\hat{\boldsymbol{w}}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n \hat{\boldsymbol{w}}^T \hat{\boldsymbol{x}}_n}\right) \tag{E4.2}$$

with respect to parameter $\hat{\boldsymbol{w}} \in R^{31\times 1}$ where

$$\hat{\boldsymbol{w}} = \begin{bmatrix} \boldsymbol{w} \\ b \end{bmatrix} \text{ and } \hat{\boldsymbol{x}}_n = \begin{bmatrix} \boldsymbol{x}_n \\ 1 \end{bmatrix}$$

and $\boldsymbol{x}_n$ and y_n for $n = 1, 2, \ldots, N$ are provided by dataset $\mathcal{D}$.

To apply GD algorithm for minimizing $f(\hat{\boldsymbol{w}})$, the gradient $\nabla_{\hat{w}} f(\hat{\boldsymbol{w}})$ is evaluated in closed-form as

$$\nabla_{\hat{w}} f(\hat{\boldsymbol{w}}) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n e^{-y_n \hat{\boldsymbol{w}}^T \hat{\boldsymbol{x}}_n}}{\left(1 + e^{-y_n \hat{\boldsymbol{w}}^T \hat{\boldsymbol{x}}_n}\right)} \hat{\boldsymbol{x}}_n \tag{E4.3}$$

Note that objective function $f(\hat{\boldsymbol{w}})$ is strictly convex, hence it admits a unique global minimizer and this minimizer is characterized by its gradient $\nabla_{\hat{w}} f(\hat{\boldsymbol{w}})$ being zero. The convexity of $f(\hat{\boldsymbol{w}})$ also assures that the GD algorithm is *insensitive* to the choice of initial point $\hat{\boldsymbol{w}}_0$.

(ii) If we call the subset of training samples with label "–1" class $\mathcal{N}$ and the subset of training samples with label "1" class $\mathcal{P}$, then minimizer $\{\boldsymbol{w}^*, b^*\}$ obtained from step (i) can be used to classify a new data point $\boldsymbol{x}$ outside the training data to class $\mathcal{P}$ or class $\mathcal{N}$ in accordance with

$$\begin{cases} \boldsymbol{x} \in \mathcal{N} & \text{if } \boldsymbol{w}^{*T}\boldsymbol{x} + b^* < 0 \\ \boldsymbol{x} \in \mathcal{P} & \text{if } \boldsymbol{w}^{*T}\boldsymbol{x} + b^* > 0 \end{cases} \tag{E4.4}$$

4. Procedure

4.1 From the course website download data matrices **D_bc_tr.mat** and **D_bc_te.mat**.

4.2 Follow Sec. 3.1 above to generate normalized train and test data sets and their labels.

4.3 Based on (E4.2) and (E4.3), prepare two MATLAB functions for evaluating the objective function and for its gradient, respectively.

4.4 Prepare MATLAB code to minimize $f(\hat{\boldsymbol{w}})$ in (E4.2) using the GD algorithm (see Sec. 2.3 of the course notes).

Remark

• To prepare the code, it is important to distinguish between $\boldsymbol{w}$ and $\hat{\boldsymbol{w}}$ as well as between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$.

4.5 Use a straightforward initial point $\hat{\boldsymbol{w}}_0 = \mathbf{0}$. Perform the minimization using the code prepared in Item 4.4 above.

Remarks

• To run the GD algorithm, the MATLAB function for back-tracking line search, `bt_lsearch.m`, is required. Make sure to download it into the directory where you intend to run GD.

• You may terminate the algorithm either by using a small convergence tolerance ε or set a number of iterations. For the problem at hand, it is a good idea to just let the algorithm run K iterations and stop.

4.6 Use the solution $\{\boldsymbol{w}^*, b^*\}$ obtained from Item 4.5 above to specify the classifier in (E4.4) and apply it to the test data from Item 4.2 above.

Try the cases of $K = 5$, $K = 12$, and $K = 75$. Evaluate and report, for each case, the performance of the classifier obtained in terms of (i) number of false positive (i.e. the number of patients in the test set, who were actually diagnosed negative but classified as positive); (ii) number of false negative (i.e. the number of patients in the test set, who were actually diagnosed positive but classified as negative); and (iii) rate of misclassification in percentage.

Include your MATLAB code in the lab report please.