Deep Learning PhD course: Hand-in assignment 1A

Niklas Wahlström

March 25, 2019

Due: 12th of April 2019, 23:59

In the first hand-in assignments 1A and 1B, we will arrive to the full implementation of a neural network. In this hand-in assignment we will implement the logistic regression model. Logistic regression can be seen as a one-layer neural network for binary classification. Hence, this code will serve as a good base to solve the full network implementation in hand-in assignment 1B.

1 Logistic regression with gradient descent

Consider a data set $\{\mathbf{x}_i, y_i\}_{i=1}^n$. Each input is a vector $\mathbf{x}_i = [x_{i1}, \ldots, x_{ip}]^\mathsf{T}$ and each output $y_i \in \{0, 1\}$ depending on which of the two classes data point i belongs to. We want to find a model for the class-1 probability $p_i = \Pr(y_i = 1|\mathbf{x}_i)$ using logistic regression. For one data point $i \in \{1, \ldots, n\}$, the logistic regression model can be described as

$$z_i = \sum_{j=1}^{p} w_j x_{ij} + b = \mathbf{w}^\mathsf{T}\mathbf{x}_i + b, \tag{1a}$$

$$p_i = \text{sigmoid}(z_i) = \frac{1}{1 + e^{-z_i}}, \tag{1b}$$

where the weight vector $\mathbf{w} = [w_1, \ldots, w_p]^\mathsf{T}$ and the offset b are the parameters. The cost J is computed by summing the following loss over all training data points

$$L_i = -y_i \ln(p_i) - (1 - y_i)\ln(1 - p_i), \tag{1c}$$

$$J = \frac{1}{n}\sum_{i=1}^{n} L_i. \tag{1d}$$

To train this model, we need access to the gradient of the cost function with respect to the model parameters, i.e. $\frac{\partial J}{\partial w_1}, \ldots, \frac{\partial J}{\partial w_p}$, and $\frac{\partial J}{\partial b}$, which you will derive below.

Exercise 1.1 *Based on the model in* (1)*, derive expressions for*

$$\frac{dJ}{db} \quad \textit{and} \quad \frac{dJ}{dw_j} \quad \textit{in terms of} \quad \frac{\partial J}{\partial z_i}, \quad \frac{dz_i}{db}, \quad \textit{and} \quad \frac{dz_i}{dw_j}. \tag{2}$$

Exercise 1.2 *Based on the model in* (1)*, derive expressions for*

$$\frac{\partial J}{\partial z_i}, \quad \frac{dz_i}{db}, \quad \textit{and} \quad \frac{dz_i}{dw_j}. \tag{3}$$

Exercise 1.3 *Implement a logistic regression model that can be trained with gradient descent based on some training dataset* $\{\mathbf{x}_i, y_i\}_{i=1}^n$.

The solution should involve the following functionalities:

- **Initialize** Write a function that initializes the parameters $\mathbf{w}$ and b. For logistic regression it is sufficient to initialize all parameters with zeros.
- **Compute cost and gradients** Compute the cost J and the gradient of the cost function with respect to all parameters $\frac{\partial J}{\partial b}$ and $\frac{\partial J}{\partial \mathbf{w}}$. For this, you need the mathematical expression in (1), (2) and (3).

- **Optimize** Update the parameters iteratively with gradient descent.
- **Predict** Using the trained model, predict $\hat{y}_i \in \{0, 1\}$ based on the corresponding input $\mathbf{x}_i$.

Exercise 1.4 *Evaluate the model on a biopsy dataset from breast cancer patient*[1] [2]*. Your task is to train a model and predict if a certain biopsy is "benign" or "malignant".*

More specifically, your tasks are:

1. Load the data `biopsy.csv`.
2. Remove all lines with `NA` values.
3. Split the dataset into 300 data points for training data and the remaining part of the dataset as test data.
4. Train a logistic regression model that minimizes the cost on training data.
5. Evaluate the performance on test data.

You should be able to reach around 95% prediction accuracy on test data. If you are running python, we have a script on the course homepage which do step 1-3. In the report, include a plot of the cost both on training and test data versus iterations. Also, include a plot with the classification accuracy, also evaluated on both test and training data.

[1] Download: `https://vincentarelbundock.github.io/Rdatasets/csv/MASS/biopsy.csv`
[2] Info: `https://vincentarelbundock.github.io/Rdatasets/doc/MASS/biopsy.html`