

"Machine Learning and Computational Statistics"

8th Homework

Exercise 1(*):

Suppose you are given a data set $Y = \{(y_i, \mathbf{x}_i), i=1, \dots, N\}$ where $y_i \in \{0, 1\}$ is the **class label** for vector $\mathbf{x}_i \in \mathbb{R}^l$. **Extract** the **gradient descent logistic regression classifier** for the two-class case (write in detail the algebraic manipulations using the hints in the relevant slides of its presentation).

Exercise 2:

Suppose you are given a data set $Y = \{(y_i, \mathbf{x}_i'), i=1, \dots, N\}$ where $y_i \in \{0, 1\}$ is the **class label** for vector $\mathbf{x}_i' \in \mathbb{R}^l$. Assume that y and \mathbf{x}' are related via the following model: $y = f(\boldsymbol{\theta}^T \mathbf{x}' + \theta_0)$, where $\boldsymbol{\theta}$ and θ_0 are the model parameters and $f(z) = 1/(1 + \exp(-az))$.

- (a) **Plot** the function $f(z)$ for various values of the parameter a .
- (b) Propose a **gradient descent scheme** to **train** this model (that is, to estimate the values of the involved parameters), based on the **minimization** of the **sum of error squares criterion**, using Y .
- (c) Can the model ever respond with a **"clear" 1** or a **"clear" 0**, for a given \mathbf{x} ?
- (d) How can we interpret the response of the model for a given \mathbf{x} ?
- (e) Propose a way for leading the model responses **very close** to **1** (for class 1 vectors) or **0** (for class 0 vectors).

Hints:

- (a) Use a more compact notation by setting $\mathbf{x}_i = [1 \ \mathbf{x}_i']^T$, $i=1, \dots, N$, and $\boldsymbol{\theta} = [\theta_0 \ \boldsymbol{\theta}]^T$. The model then becomes $y = f(\boldsymbol{\theta}^T \mathbf{x})$.
- (b) The sum of error squares criterion in this case is $J(\boldsymbol{\theta}) = \sum_{n=1}^N (y_n - f(\boldsymbol{\theta}^T \mathbf{x}_n))^2$.
- (c) It is $f'(z) = \frac{df(z)}{dz} = af(z)(1 - f(z))$.

Exercise 3:

Consider a **two-class one-dimensional** classification problem and suppose you are given a data set

$$Y = \{(y_i, x_i), x_i \in \mathbb{R}, y_i \in \{-1, +1\}, i = 1, \dots, N\}.$$

A classifier for this problem is of the form

$$\text{"Assign a given } x \text{ to class } +1 \text{ (} -1 \text{) if } x > T \text{ (} < T \text{)"}.$$

Propose an (efficient) algorithm for determining the threshold T that borders the two classes, so that to minimize the classification error, **based on** the data set Y .

Hints:

- (a) Exploit the fact that x_i 's are **ordered** in the line of the real numbers.
- (b) For your help and in order to get a visual inspection, consider a specific data set, e.g., $Y = \{ \dots, (-1, -5), (-1, -3), (-1, 0), (+1, -1), (+1, 1), (+1, 3), \dots \}$ and mark on the line of the real numbers the data points, using different colors for points from different classes.

Exercise 4 (python code + text):

Consider a two-class, two-dimensional classification problem for which you can find attached two **sets**: one for **training** and one for **testing** (file [HW8.mat](#)). Each of these sets consists of pairs of the form (y_i, x_i) , where y_i is the **class label** for vector x_i . Let N_{train} and N_{test} denote the number of training and test sets, respectively. The data are given via the following arrays/matrices:

- **train_x** (a $N_{train} \times 2$ **matrix** that contains in its **rows** the **training** vectors x_i)
- **train_y** (a N_{train} -dim. column **vector** containing the **class labels** (1 or 2) of the corresponding **training** vectors x_i included in **train_x**).
- **test_x** (a $N_{test} \times 2$ **matrix** that contains in its **rows** the **test** vectors x_i)
- **test_y** (a N_{test} -dim. column **vector** containing the **class labels** (1 or 2) of the corresponding **test** vectors x_i included in **test_x**).

Assume that the two classes, ω_1 and ω_2 are modeled by **normal distributions**.

- (a) Adopt the **Bayes classifier**.
 - i. Use the training set to **estimate** $P(\omega_1)$, $P(\omega_2)$, $p(x|\omega_1)$, $p(x|\omega_2)$ (Since $p(x|\omega_j)$ is modeled a normal distribution, it is completely identified by μ_j and Σ_j . Use the **ML estimates** for them as given in the lecture slides).
 - ii. **Classify** the points x_i of the test set, using the **Bayes classifier** (for each point apply the Bayes classification rule and keep the class labels, to an a N_{test} -dim. column **vector** , called **Btest_y** containing the **estimated class labels** (1 or 2) of the corresponding **test** vectors x_i included in **test_x**).
 - iii. Estimate the error classification probability ((1) **compare** **test_y** and **Btest_y** , (2) **count** the positions where both of them have the same class label and (3) **divide** with the total number of test vectors).

- (b) Adopt the **naïve Bayes classifier**.

○ ○

Recall that $x = [x_1, x_2]^T$

- i. Use the training set to estimate $P(\omega_1)$, $P(\omega_2)$, $p(x_1|\omega_1)$, $p(x_2|\omega_1)$, $p(x_1|\omega_2)$, $p(x_2|\omega_2)$ (Each $p(x|\omega_j)$ is written as $p(x|\omega_j) = p(x_1|\omega_j) * p(x_2|\omega_j)$). Use the **ML estimates** of the mean and variance for each one of the 1-dim. pdfs).
 - ii. Classify the points $\mathbf{x}_i = [x_{i1}, x_{i2}]^T$ of the test set, using the naïve Bayes classifier (Estimate $p(\mathbf{x}|\omega_j)$ with $p(x_{i1}|\omega_j) * p(x_{i2}|\omega_j)$ and then apply the Bayes rule. Keep the class labels, to an a N_{test} -dim. column **vector** , called **NBtest_y** containing the **estimated class labels** (1 or 2) of the corresponding **test** vectors \mathbf{x}_i included in **test_x**)
 - iii. Estimate the error classification probability (work as in the previous case).
- (c) Adopt the **k-nearest neighbor classifier**, for $k = 5$ and estimate the classification error probability.
 - (d) Adopt the **logistic regression classifier** and (i) train it using the training set and then (ii) measure its performance on the test set.
 - (e) Depict graphically the training set, using different colors for points from different classes.
 - (f) Report the classification results obtained by the four classifiers and comment on them. Under what conditions, the first two classifiers would exhibit the same performance?

Hint: Use the attached Python code.