Introduction to Machine Learning
Homework 2

吴紫航 171860659

1 [30pts] Multi-Label Logistic Regression

In multi-label problem, each instance $\boldsymbol{x}$ has a label set $\boldsymbol{y} = \{y_1, y_2, ..., y_L\}$ and each label $y_i \in \{0, 1\}, \forall 1 \leq i \leq L$. Assume the post probability $p(\boldsymbol{y} \mid \boldsymbol{x})$ follows the conditional independence:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{i=1}^{L} p(y_i \mid \boldsymbol{x}). \tag{1.1}$$

Please use the logistic regression method to handle the following questions.
(1) [15pts] Please give the log-likelihood function of your logistic regression model;

解: 设$\boldsymbol{\omega} = (\boldsymbol{\omega_1}; \boldsymbol{\omega_2}; \boldsymbol{\omega_3}; \ ... \ ; \boldsymbol{\omega_L})$ $\quad \boldsymbol{b} = (b_1; b_2; b_3; \ ... \ ; b_L)$

由对数几率回归

$p(y = 1|\boldsymbol{x}) = \frac{e^{\boldsymbol{\omega}^T \boldsymbol{x}+b}}{1+e^{\boldsymbol{\omega}^T \boldsymbol{x}+b}}$

$p(y = 0|\boldsymbol{x}) = \frac{1}{1+e^{\boldsymbol{\omega}^T \boldsymbol{x}+b}}$

给定数据集$\{(\boldsymbol{x_i}, \boldsymbol{y_i})\}_{i=1}^m$, log-likelihood function为

$l(\boldsymbol{\omega}, \boldsymbol{b}) = \sum_{i=1}^{m} ln \ p(\boldsymbol{y_i}|\boldsymbol{x_i}; \boldsymbol{\omega}, \boldsymbol{b})$

$= \sum_{i=1}^{m} ln \ \prod_{j=1}^{L} \ p(y_{ij}|\boldsymbol{x_i}; \boldsymbol{\omega_j}, b_j)$

$= \sum_{i=1}^{m} \sum_{j=1}^{L} ln \ p(y_{ij}|\boldsymbol{x_i}; \boldsymbol{\omega_j}, b_j)$

为了便于讨论，令$\boldsymbol{\beta} = (\boldsymbol{\beta_1}; \boldsymbol{\beta_2}; ...; \boldsymbol{\beta_L}), \boldsymbol{\beta_j} = (\boldsymbol{\omega_j}; b_j), j = 1, 2, ..., L$

$\hat{\boldsymbol{x}} = (\boldsymbol{x}, 1)$,则$\boldsymbol{\omega_j}^T \boldsymbol{x} + b_j = \boldsymbol{\beta_j} \hat{\boldsymbol{x}}$

再令$p1(\hat{\boldsymbol{x}}; \boldsymbol{\beta_j}) = p(y_j = 1| \hat{\boldsymbol{x}}; \boldsymbol{\beta_j}) \ \ p0(\hat{\boldsymbol{x}}; \boldsymbol{\beta_j}) = p(y_j = 0| \hat{\boldsymbol{x}}; \boldsymbol{\beta_j})$

则重写似然项为$p(y_{ij}|\boldsymbol{x_i};\boldsymbol{\omega_j},b_j) = y_{ij}p1(\hat{\boldsymbol{x_i}};\boldsymbol{\beta_j}) + (1-y_{ij})p0(\hat{\boldsymbol{x_i}};\boldsymbol{\beta_j})$
故等价于最小化
$l(\boldsymbol{\beta}) = \sum_{i=1}^{m}\sum_{j=1}^{L}(-y_{ij}\boldsymbol{\beta_j^T}\hat{\boldsymbol{x_i}} + ln(1+e^{\boldsymbol{\beta_j^T}\hat{\boldsymbol{x_i}}}))$
注：更严格的形式可以考虑，对每个标签取平均值,
即$l(\boldsymbol{\beta}) = \frac{1}{L}\sum_{i=1}^{m}\sum_{j=1}^{L}(-y_{ij}\boldsymbol{\beta_j^T}\hat{\boldsymbol{x_i}} + ln(1+e^{\boldsymbol{\beta_j^T}\hat{\boldsymbol{x_i}}}))$

(2) [15pts] Please calculate the gradient of your log-likelihood function and show the parameters updating step using gradient descent.
解：由凸优化理论
$\boldsymbol{\beta^*} = \underset{\boldsymbol{\beta}}{arg\,min}\, l(\boldsymbol{\beta})$
$\boldsymbol{\beta^{t+1}} = \boldsymbol{\beta^t} - \gamma\nabla l(\boldsymbol{\beta}) = \boldsymbol{\beta^t} + \gamma\sum_{i=1}^{m}\sum_{j=1}^{L}\hat{\boldsymbol{x_i}}(y_{ij} - p1(\hat{\boldsymbol{x_i}};))$

2 [70pts] Logistic Regression from scratch

Implementing algorithms is a good way of understanding how they work in-depth. In case that you are not familiar with the pipeline of building a machine learning model, this article can be an example (link).

In this experiment, you are asked to build a classification model on one of UCI data sets, Letter Recognition Data Set (click to download). In particular, the objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The detailed statistics of this data set is listed in Table 1. The data set was then randomly split into train set and test set with proportion 7 : 3. Also, letters from 'A' to 'Z' are mapped to digits '1' to '26' respectively as represented in the last column of the provided data set.

Table 1: Statistics of the data set.

Property	Value	Description
Number of Instances	20,000	Rows of the data set
Number of Features	17	Columns of the data set
Number of classes	26	Dimension of the target attribute

In order to build machine learning models, you are supposed to implement Logistic Regression (LR) algorithm which is commonly used in classification tasks. Specifically, in this experiment, you have to adapt the traditional binary class LR method to tackle the multi-class learning problem.

(1) [**10pts**] You are encouraged to implement the code using *Python3* or *Matlab*, implementations in any other programming language will not be graded. Please name the source code file (which contains the main function) as *LR_main.py* (for python3) or *LR_main.m* (for matlab). Finally, your code needs to print the testing performance on the provided test set once executed.

(2) [**30pts**] Functions required to implement:

- Implement LR algorithm using gradient descent or Newton's method.
- Incorporate One-vs-Rest (OvR) strategy to tackle multi-class classification problem.

(3) [**30pts**] Explain implementation details in your submitted report (source code should not be included in your PDF report), including optimization details and hyper-parameter settings, etc. Also, testing performance with respect to Accuracy, Precision, Recall, and F_1 score should be reported following the form of Table 2.

Table 2: Performance of your implementation on test set.

Performance Metric	Value (%)
accuracy	00.00
micro Precision	00.00
micro Recall	00.00
micro F_1	00.00
macro Precision	00.00
macro Recall	00.00
macro F_1	00.00

NOTE: Any off-the-shelf implementations of LR or optimization methods are **NOT ALLOWED** to use. When submitting your code and report, all files should be placed in the same directory (without any sub-directory).

注:详细过程见报告report.pdf和代码LR_main.py