The Chinese University of Hong Kong, Shenzhen
School of Data Science

DDA 3020 · Homework 2
Due: 23:59, March 24th, 2024

Instructions:

- This assignment accounts for 15/100 of the final score.
- You must independently complete each assignment.
- Late submission will get discounted score: 20 percent discount on $(0, 24]$ hours late; 50 percent discount on $(24, 120]$ hours late; no score on late submission of more than 120 hours.

---

1 Written Problems (50 pts.)

1.1 Problem: Overfitting and Regularized Logisitc Regression (5 pts.)

1) (2 pts.) Plot the sigmoid function $1/(1+\exp^{-wX})$ for increasing weights $w \in \{1, 5, 100\}$ with $X \in \mathbb{R}$. A qualitative sketch will suffice. Utilize these plots to explain why large weights can lead to overfitting in logistic regression.
2) (3 pts.) To mitigate overfitting, it is preferable to have smaller weights. To accomplish this, rather than utilizing maximum conditional likelihood estimation M(C)LE for logistic regression:

$$\max_{w_0,\cdots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d), \tag{1}$$

we can consider maximum conditional a posterior M(C)AP estimation:

$$\max_{w_0,\cdots,w_d} \prod_{i=1}^{n} P(Y_i|X_i, w_0, \cdots, w_d)P(w_0, \cdots, w_d), \tag{2}$$

where $P(w_0, \cdots, w_d)$ is a prior on the weights.
Given a standard Gaussian prior $\mathcal{N}(0, \mathbf{I})$ for the weight vector w, please derive the **gradient ascent** update rules for the weights and explain why M(C)AP can address overfitting issue.

1.2 Problem: Multi-class Logisitc Regression (16 pts.)

In this question, we will derive the "multi-class logistic regression" algorithm, assuming the dataset $\mathcal{D}$ is d-dimensional (with d features) and contains n entries.
Given a training set $\{(x^i, y^i)|i = 1, \cdots, n\}$ where $x^i \in \mathbb{R}^{d+1}$ is a feature vector and $y^i \in \mathbb{R}^k$ is a one-hot encoded binary vector with k entries representing classes. In a one-hot vector, the corresponding class label is 1, and all other entries are 0s. For example, if the label of x^i is 3, then the corresponding y^i should be $[0, 0, 1, \cdots, 0] \in \mathbb{R}^k$.

We aim to determine the parameters $\hat{w} \in \mathbb{R}^{k\times(d+1)}$ (representing one weight vector for each class) that maximize the likelihood for the training set, given a parametric model in the following form:

$$p(y_c^i = 1|x^i; w) = \frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}. \tag{3}$$

Note that $\frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}$ is always between 0 and 1, and $\sum_c p(y_c^i = 1|x^i; w)$ is always 1, which are desired properties of a probability distribution. Therefore, $\frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}$ is also known as the softmax function.
Since we know the probability sums to 1, we don't care about predicting the probability of the last (k^{th}) class, since we can calculate $p(y_k^i = 1|x^i; w)$ by:

$$p(y_k^i = 1|x^i; w) = 1 - \frac{\sum_{c'=1}^{k-1} \exp(w_{c'}^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)}. \tag{4}$$

1) (4 pts.) Show the equivalence between the Eq. (5) and Eq. (6). Provide a short justification for why each line follows from the previous one in your derivation. (This is how we store less weights by making use of the fact that the probabilities sum to 1).

$$p(y_c^i = 1|x^i; w) = \frac{\exp(w_c^\top x^i)}{\sum_{c'} \exp(w_{c'}^\top x^i)} \tag{5}$$

$$= \begin{cases} \dfrac{\exp(w_c^\top x^i)}{1 + \sum_{c'=1}^{k-1} \exp(w_c^\top x^i)}, & \text{if } c < k \\ \dfrac{1}{1 + \sum_{c'=1}^{k-1} \exp(w_c^\top x^i)}, & \text{if } c = k \end{cases} \tag{6}$$

2) (4 pts.) Derive the conditional log likelihood for softmax regression. For the sake of simplicity, we only consider Eq. (5) as $p(y_c^j|x^j, w)$. Show the equivalence between the Eq. (7) and Eq. (8). Provide a short justification for why each line follows from the previous one in your derivation.

$$\mathcal{L}(w) \equiv \ln \prod_{j=1}^{n} p(y_c^j|x^j, w) \text{ [here } c \text{ is the true class of } x^j\text{]} \tag{7}$$

$$= \sum_{j=1}^{n} \sum_{c=1}^{k} \left[y_c^j (w_c^\top x^j) - y_c^j \ln \left(\sum_{c'} \exp(w_{c'}^\top x^j) \right) \right]. \tag{8}$$

3) (4 pts.) Next, we will derive the gradient of the previous expression with respect to the c^{th} class of the weight matrix w_c, i.e., $\frac{\partial \mathcal{L}(w)}{\partial w_c}$, where $\mathcal{L}(w)$ denotes the log likelihood from Eq. (8). We will perform a few steps of the derivation, and then ask you to do one step at the end. If we task the derivative of Eq. (8) with respect to w_c, we get the following experssion:

$$\nabla_{w_c} \mathcal{L}(w) = \nabla_{w_c} \sum_{j=1}^{n} \sum_{c=1}^{k} \left[y_c^j (w_c^\top x^j) - y_c^j \ln \left(\sum_{c'} \exp(w_{c'}^\top x^j) \right) \right]. \tag{9}$$

The blue expression is linear in w_c, so it can be simplified to $\sum_{j=1}^{n} y_c^j x^j$. For the red expression,first we consider a fixed $j \in [1, n]$. Use the chain rule to verify that

$$\nabla_{w_c} \sum_{c=1}^{k} y_c^j \ln \left(\sum_{c'} \exp(w_{c'}^\top x^j) \right) \tag{10}$$

$$= \frac{\exp(w_c^\top x^j)}{\sum_{c'} \exp(w_{c'}^\top x^j)} x^j. \tag{11}$$

4) (2 pts.) Now use 11 (and the previous discussion) to show that overall, Eq. (9), i.e., $\nabla_{w_c} \mathcal{L}(w)$, is equal to

$$\nabla_{w_c} \mathcal{L}(w) = \sum_{j=1}^{n} x^j (y_c^j - p(y_c^j = 1 | x^j; w)). \tag{12}$$

5) (2 pts.) Since the log likelihood is concave, it is easy to optimize using gradient ascent. Derive the update rule for **gradient ascent** with respect to learning rate for w_c w.r.t. η, y^j, x^j, and $p(y^j = 1 | x^j; w^{(t)})$. Feel free to index into the vectors using subscripts.

1.3 Problem: Support Vector Machine 1 (6 pts.)

Given a binary data set:
Class -1: $\{(0,1), (\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}), (-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})\}$. Class +1: $\{(0, \sqrt{2}), (1, -1), (-1, -1)\}$.
1) (3 pts.) Could you find an SVM classifier (without slack variable) for this dataset? Please explain your reasoning, possibly with the help of a plot sketch.
2) (3 pts.) Use SVM by expanding the original feature vector $x = [x_1, x_2]$ to $x = [x_1^2, x_2^2]$, find the svm of this given data set and predict the label of $(-\frac{1}{2}, \sqrt{2})$.

1.4 Problem: Support Vector Machine 2 (18 pts.)

In this question you have to derive the dual form of SV regression (SVR). Your training data is $\{(x_1, y_1), \cdots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$.
Since the hinge loss that we used in class is only designed for classification we cannot use that for regression. A frequently used loss function for regression is the epsilon sensitive loss:

$$\mathcal{L}_\epsilon(x, y, f) = |y - f(x)|_\epsilon = \max(0, |y - f(x)| - \epsilon). \tag{13}$$

Here x is the input, y is the output, and f is the function used for predicting the label. Using this notation, the SVR cost function is defined as

$$\frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \mathcal{L}_\epsilon(x_i, y_i, f), \tag{14}$$

where $f(x) = w^\top x$, and C, $\epsilon > 0$ are parameters.
1) (3 pts.) Introduce appropriate slack variables, and rewrite this problem as a quadratic problem (i.e. quadratic objective with linear constraints). This form is called the Primal form of support vector regression.
2) (2 pts.) Write down the Lagrangian function for the above primal form.

3) (3 pts.) Using the Karush Kunh Tucker conditions derive the dual form.
4) (1 pts.) Can we use quadratic optimization solvers to solve the dual problem?
5) (2 pts.) How would you define support vectors in this problem?
6) (2 pts.) Write down the equation that can be used for predicting the label of unseen sample X.
7) (1 pts.) Is it possible to kernelize this algorithm?
8) (2 pts.) What happens if we change ϵ?
9) (2 pts.) What happens if we change C?

1.5 Problem: C4.5 Decision Tree (5 pts.)

In this problem, we are set to construct a decision tree using the C4.5 algorithm. This particular algorithm leverages the Information Gain Ratio as a construction principle, guiding the formation of the decision tree.

Let's denote our dataset as D and its size as $|D|$. Suppose the dataset contains K distinct classes, labeled as C_k, where k ranges from 1 to K. The size of the individual data points allocated within each class C_k is expressed as $|C_k|$. As such, the sum of the sizes of all classes, i.e. $\sum_{k=1}^{K} |C_k| = |D|$. Given that the attribute A comprises n unique values, denoted as $\{a_1, a_2, \cdots, a_n\}$, it can bifurcate the dataset D into n subsets $D_1, D_2, \cdots, D_n$. Each subset size is in turn represented as $|D_i|$. It follows that the aggregation of all these subset sizes, i.e. $\sum_{i=1}^{n} |D_i|$, equates to the overall size of the original dataset, that is, $|D|$. Let the set of samples in subset D_i that belong to class C_k be represented by D_{ik}, that is, $D_{ik} = D_i \cap C_k$. The count of samples in D_{ik} is represented by $|D_{ik}|$. The empirical entropy of dataset is:

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}.$$

The empirical conditional entropy of attribute A respect to dataset D is:

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}.$$

The information gain is:

$$g(D, A) = H(D) - H(D|A).$$

We can calculate the Information Gain Ratio of attribute A for the dataset D is:

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}.$$

We can utilize the Information Gain Ratio to build a decision tree through the C4.5 algorithm, as described in Algorithm 1. For more information about the C4.5 algorithm, you can refer to the example `https://sefiks.com/2018/05/13/a-step-by-step-c4-5-decision-tree-example/`. According to the training dataset provided Table 1, generate a decision tree using the Information Gain Ratio (C4.5 algorithm).

Algorithm 1 C4.5 Algorithm

Require: Training dataset D, attribute set A, threshold ϵ

Ensure: Decision tree T

1: If D belongs to the same class C_k, T is a single-node tree, and class C_k is assigned as the label of the node. Return T

2: If A is an empty set, set T as a single-node tree, and assign the label of the node as the class with the highest number of instances. Return T

3: Calculate g_R for each attribute A, select the attribute A_g with the maximum **information gain ratio**

4: If the **information gain ratio** of A_g is less than ϵ, T is a single-node tree, and the class C_k with the maximum number of instances in D is assigned as the label. Return T

5: Split A_g into several non-empty subsets D_i

6: For each subset D_i, use it as the training dataset, $A-\{A_g\}$ as the attribute set, and recursively call the previous steps to obtain T_i. Return T_i

ID	Age	Work	House	Credit	Class
1	young	No	No	Normal	No
2	young	No	No	Good	No
3	young	Yes	No	Good	Yes
4	young	Yes	Yes	Normal	Yes
5	young	No	No	Normal	No
6	middle	No	No	Normal	No
7	middle	No	No	Good	No
8	middle	Yes	Yes	Good	Yes
9	middle	No	Yes	Excellent	Yes
10	middle	No	Yes	Excellent	Yes
11	old	No	Yes	Excellent	Yes
12	old	No	Yes	Good	Yes
13	old	Yes	No	Good	Yes
14	old	Yes	No	Excellent	Yes
15	old	No	No	Normal	No

Table 1: Dataset of Loan Application

2 Coding Problems (50 pts.)

This part consists of two main tasks: Logistic Regression (25 pts.) and Support Vector Machine (25 pts.), aimed at deepening your practical understanding of these algorithms. You will find two files, "Logistic_Regression.ipynb" and "Support_Vector_Machine.ipynb." Your objective is to fill in the code in the **TASK** blank to complete the algorithm using the given dataset. Kindly note that the "weatherAUS.csv" dataset is suitable for a Logistic Regression task, while "pulsar_stars.csv" is better suited for a Support Vector Machine task.

Submission Requirement

- Your completed two jupyter notebook files, naming "StudentID_Logisitc_Regression.ipynb" and "StudentID_Support_Vector_Machine.ipynb". Your jupyter notebook files should **contain the running output** of each step (numbers, plots, etc.). If your notebook has only code but no output results, you will get a discounted score.
- Detailed report named "StudentID_homework2.pdf" on the problems in the writing part. Ensure that you include a comprehensive description of your solutions to these problems. Handwritten is also acceptable.
- Submission list: 1 pdf report for the writing part and 2 jupyter notebook for coding part.