OSU CSE 3521
Homework #4: Problem Set

Release Date: November 23rd, 2021

Submission Instructions

Due Date: December 8st (23:59 ET), 2021

Submission: Please submit your solutions in a single PDF file named HW4_name_number.pdf (e.g., HW4_chao_209.pdf) to Carmen. You may write your solutions on paper and scan it, or directly type your solutions and save them as a PDF file. *Submission in any other format will not be graded.*

We highly recommend that you write down the derivation of your answers, and highlight your answers clearly!

Collaboration: You may discuss with your classmates at a very high level. However, you need to write your own solutions and submit them separately by yourself. Also in your written report, you need to list with whom you have discussed for each problem (please do so in the first page). Please consult the syllabus for what is and is not an acceptable collaboration.

Calculation: Please perform rounding to your results after the second decimal number, *unless stated otherwise*. For example, 1.245 becomes 1.25 and -1.228 becomes -1.23.

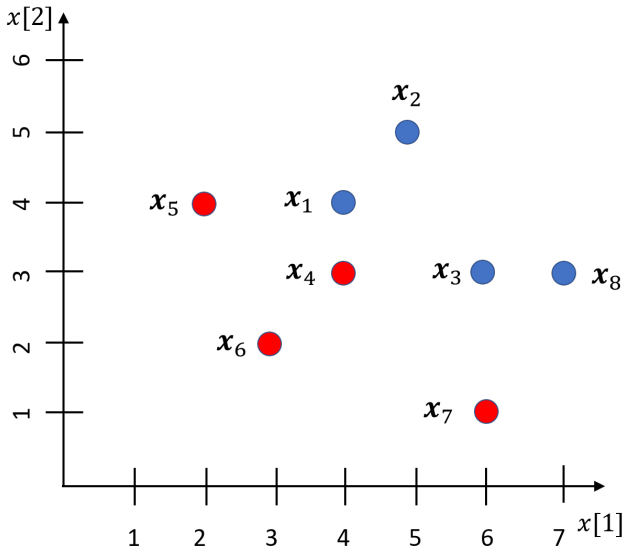

Figure 1: A two-dimensional, two-class data set of 8 data instances

1 Cross-validation for K nearest neighbor (KNN) classifiers [10 points]

Figure 1 shows a data set of 8 data instances, each of them is two-dimensional. Every point's coordinates are both integer values: $\boldsymbol{x}_1 = (4,4), \boldsymbol{x}_2 = (5,5), \boldsymbol{x}_3 = (6,3), \boldsymbol{x}_4 = (4,3), \boldsymbol{x}_5 = (2,4), \boldsymbol{x}_6 = (3,2), \boldsymbol{x}_7 = (6,1), \boldsymbol{x}_8 = (7,3)$. There are two classes denoted by either blue or red color. In the following, you will practice cross-validation for KNN, using the Euclidean distance.

Specifically, you will consider a special case, called *leave-one-out* cross-validation. The idea is to separate your training data (8 data points) into 8 folds (each with one unique data point), and every time treats one data example as the validation example; the others, as the training examples. For instance, let $\boldsymbol{x}_4$ be the validation example, and the other 7 data points as the training examples, then the 1-NN prediction of $\boldsymbol{x}_4$ will be *blue* since $\boldsymbol{x}_1$ is the closest point to it (i.e., nearest neighbor) but the true label of $\boldsymbol{x}_4$ (i.e., y_4) is *red*.

Let us now denote by D the whole data set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{8}$, and let us denote by $D \setminus \{(x_j, y_j)\}$ the data set of 7 points after removing (x_j, y_j). Here, y_i is the true label of each data point (i.e., the color in Figure 1). Let $\hat{y}_j = f_{KNN}(x_j; D \setminus \{(x_j, y_j)\})$ denote the KNN prediction of x_j, using $D \setminus \{(x_j, y_j)\}$ as the training data. The leave-one-out (LOO) cross-validation accuracy can be

represented as

$$\text{LOO-Accuracy} = \frac{1}{8} \sum_{j \in \{1,\cdots,8\}} \mathbf{1}[y_j == f_{KNN}(x_j; D \setminus \{(x_j, y_j)\})], \qquad (1)$$

where $\mathbf{1}[\cdot]$ is the indicator function with $\mathbf{1}[True] = 1$; otherwise, 0.

1. **[5 points]** Please compute the LOO-Accuracy for the 1-NN classifier. That is, the prediction of $\boldsymbol{x}_j$ is the true label of its nearest neighbor in $D \setminus \{(x_j, y_j)\}$.

2. **[5 points]** Please compute the LOO-Accuracy for the 3-NN classifier. That is, the prediction of $\boldsymbol{x}_j$ is the major label of its 3 nearest neighbors in $D \setminus \{(x_j, y_j)\}$.

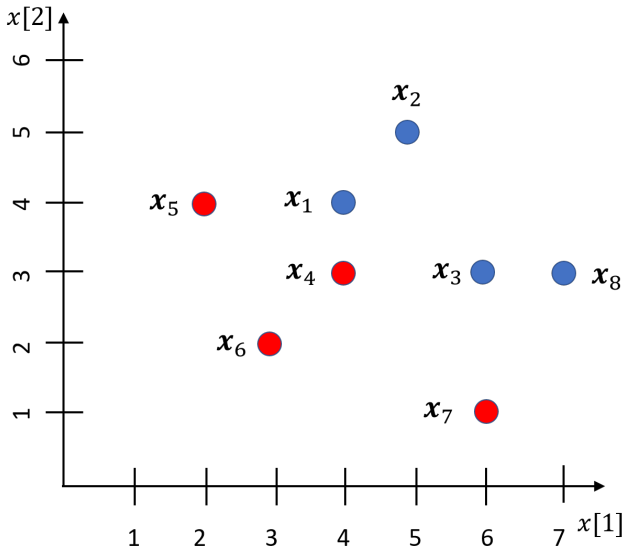

Figure 2: A two-dimensional, two-class data set of 8 data instances

2 Decision trees [14 points]

Figure 2 shows a data set of 8 data instances, each of them is two-dimensional. Every point's coordinates are both integer values: $\boldsymbol{x}_1 = (4,4), \boldsymbol{x}_2 = (5,5), \boldsymbol{x}_3 = (6,3), \boldsymbol{x}_4 = (4,3), \boldsymbol{x}_5 = (2,4), \boldsymbol{x}_6 = (3,2), \boldsymbol{x}_7 = (6,1), \boldsymbol{x}_8 = (7,3)$. There are two classes denoted by either blue or red color. For brevity, let us index class blue as class 1, class red as class 2.

In this question, you will practice how to construct a decision tree using the Gini impurity. For example, if you want to find a threshold $\tau\star$ on $x[1]$ to separate the 8 points into two parts, you are to solve the following optimization problem

$$\tau\star = \arg\min_{\tau} \sum_{c\in\{1,2\}} P'_c \times (1 - P'_c) + \sum_{c\in\{1,2\}} P''_c \times (1 - P''_c), \tag{2}$$

where $P'_c = \frac{|\{x_i[1]>\tau \text{ AND } y_i=c\}|}{|\{x_i[1]>\tau\}|}$ and $P''_c = \frac{|\{x_i[1]\leq\tau \text{ AND } y_i=c\}|}{|\{x_i[1]\leq\tau\}|}$. That is, P'_c is the proportion of points of class c within those whose $x[1] > \tau$; P''_c is the proportion of points of class c within those whose $x[1] \leq \tau$. For example, if $\tau = 2.5$, then $P'_1 = \frac{4}{7}$ while $P''_1 = \frac{0}{1}$. You are to compare among different possible τ to find the $\tau\star$ that minimize Equation 2.

Specifically, for this question, you are to construct the decision tree shown in Figure 3. That is, you are to fill in the thresholds of nodes A, B, and C, and the colors (majority colors of that node) of nodes D, E, F, and G.

Here are several rules and hints to follow:

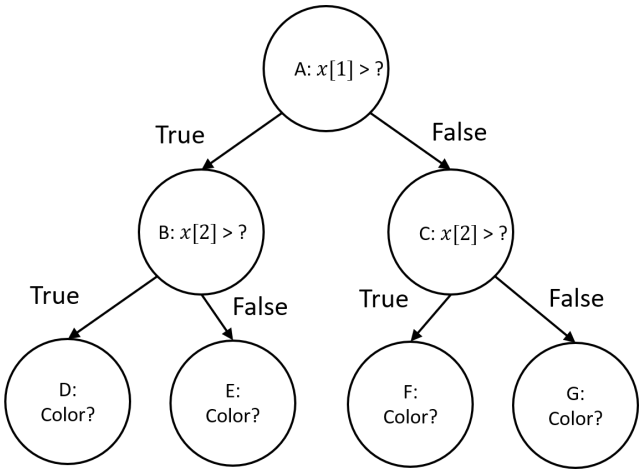

Figure 3: A decision tree of 7 nodes.

- You only need to consider the following possible $\tau \in \{2.5, 3.5, 4.5, 5.5, 6.5\}$ for $x[1]$ and $\tau \in \{1.5, 2.5, 3.5, 4.5\}$ for $x[2]$.
- You must select a τ that can separate the points into two parts.
- If you are to apply Equation 2 to find a threshold on $x[2]$, you simply change the definitions of P_c' and P_c'' to separate points by $x[2]$ rather than by $x[1]$.
- For nodes B and C, where only a part of the 8 points will fall into, P_c' and P_c'' are only computed for those points. For instance, if A is $x[1] > 5.5$, then for node B, P_c' and P_c'' are only computed on $\{\boldsymbol{x}_3, \boldsymbol{x}_7, \boldsymbol{x}_8\}$.

Please solve the following two questions.

1. **[7 points]** Fill in the thresholds of nodes A, B, and C following the Gini impurity (i.e., Equation 2). Fill in the colors (majority colors of that node) of nodes D, E, F, and G.
2. **[7 points]** Now, let us modify Equation 2 a bit to the following form

$$\tau\star = \arg\min_\tau \max \left\{ \sum_{c\in\{1,2\}} P_c' \times (1 - P_c'), \sum_{c\in\{1,2\}} P_c'' \times (1 - P_c'') \right\}, \quad (3)$$

   in which we replace summation by max. Fill in the thresholds of nodes A, B, and C following the modified Gini impurity (i.e., Equation 3). Fill in the colors (majority colors of that node) of nodes D, E, F, and G.

3 Logistic regression [10 points]

In the lectures, we showed that the loss function of logistic regression $\boldsymbol{w}^\top \boldsymbol{x} + b$ for a labeled data instance $(\boldsymbol{x}, y \in \{0, 1\})$ is

$$\ell = -y \times \log \sigma(\boldsymbol{w}^\top \boldsymbol{x} + b) - (1 - y) \times \log(1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x} + b)) \tag{4}$$

where $\sigma(\boldsymbol{w}^\top \boldsymbol{x} + b) = \frac{1}{1+\exp\{-(\boldsymbol{w}^\top \boldsymbol{x}+b)\}}$ is the sigmoid function.

Now given a 1-dimensional data point $(x = 2, y = 0)$ and suppose $w = 1$ and $b = 1$, please compute

1. **[5 points]** $\frac{\partial \ell}{\partial w}$ at $w = 1$ and $b = 1$
2. **[5 points]** $\frac{\partial \ell}{\partial b}$ at $w = 1$ and $b = 1$

Figure 4: The LibSVM Graphic Interface.

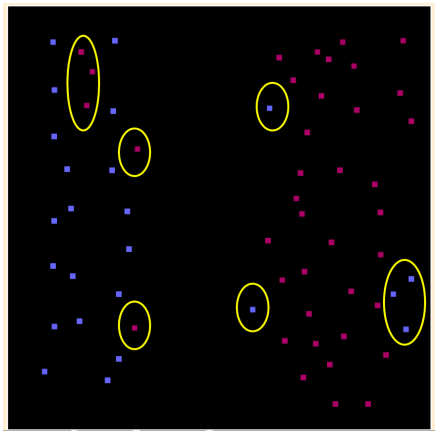

Figure 5: A two-class, two-dimensional data set. The yellow circles highlight the points that are a bit like outliers.

4 SVM [11 points]

Please go to https://www.csie.ntu.edu.tw/~cjlin/libsvm/ and use the Graphic Interface (shown in Figure 4). Please construct a data set similar to Figure 5 and play with the following options:

- -t 0 -c 100, which builds a linear classifier.
- -t 0 -c 0.1, which also builds a linear classifier.
- -t 2 -c 100, which builds a nonlinear classifier.
- -t 2 -c 10000, which also builds a nonlinear classifier.
- -t 2 -c 1000000, which also builds a nonlinear classifier.

The command $-t$ decides if the boundary is linear (0) or nonlinear (2). The command $-c$ decides how complex the boundary will be in order to classify the training data correctly (the larger, the more complex).

1. **[5 points]** Please play with the above five options and copy and paste your resulting figures (i.e., after click Run, you will see the decision boundaries being generated).
2. **[2 points]** Which option has the most misclassified training data?
3. **[2 points]** Which option has the least misclassified training data?
4. **[2 points]** Which option will you use to construct a classifier for the future test data? and why (a short paragraph)?

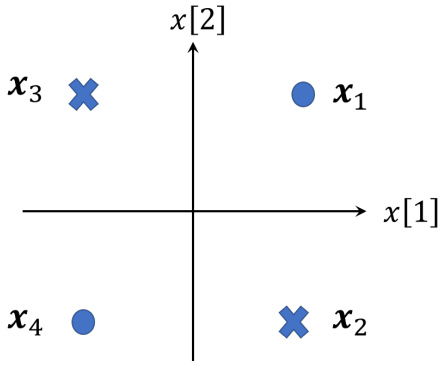

Figure 6: A two-dimensional, two-class data set of 4 data instances

5 Neural networks [15 points]

In this question, you will practice the decision rule of a simple neural network. Figure 6 shows a simple two-dimensional dataset of four data instances and two classes: crosses for class -1; circles for class $+1$. $\boldsymbol{x}_1 = [1, 1]^\top$, $\boldsymbol{x}_2 = [1, -1]^\top$, $\boldsymbol{x}_3 = [-1, 1]^\top$, and $\boldsymbol{x}_4 = [-1, -1]^\top$.

5.1 Decision with nonlinearity [10 points]

Please consider a one-hidden-layer neural network as follows. Given an input vector $\boldsymbol{x} \in \mathbb{R}^2$, this network first generates a two-dimensional vector $\boldsymbol{z} \in \mathbb{R}^2$ by $\boldsymbol{z} = \text{sign}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$, where $\boldsymbol{W}$ is a 2-by-2 matrix and $\boldsymbol{b} \in \mathbb{R}^2$. The sign function operates on each input element alone and outputs 1 if the input element is larger than 0; otherwise, -1. For example, $\text{sign}([0.5, -0.5]^\top) = [1, -1]^\top$.

Given $\boldsymbol{z} \in \mathbb{R}^2$, the network then predicts the label $\hat{y}$ by $\hat{y} = \text{sign}(\boldsymbol{u}^\top \boldsymbol{z} + a)$, where $\boldsymbol{u} \in \mathbb{R}^2$ and a is a scalar (i.e., $a \in \mathbb{R}$).

Suppose

$$\boldsymbol{W} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix},$$
$$\boldsymbol{b} = \begin{bmatrix} -1 \\ -1 \end{bmatrix},$$
$$\boldsymbol{u} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$
$$a = 1,$$

please compute $\hat{y}_1$ for $\boldsymbol{x}_1$, $\hat{y}_2$ for $\boldsymbol{x}_2$, $\hat{y}_3$ for $\boldsymbol{x}_3$, and $\hat{y}_4$ for $\boldsymbol{x}_4$. Please further compute the classification accuracy: $\frac{1}{4}\sum_{i=1}^{4} \mathbf{1}[\hat{y}_i == y_i]$.

5.2 Decision without nonlinearity [5 points]

Let us consider the same neural network as above, with one slight change. Let us remove the sign function in computing $\boldsymbol{z}$. That is, $\boldsymbol{z} = \text{sign}(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b})$ becomes $\boldsymbol{z} = (\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$. All other parameters and computations remain the same.

Please re-compute $\hat{y}_1$ for $\boldsymbol{x}_1$, $\hat{y}_2$ for $\boldsymbol{x}_2$, $\hat{y}_3$ for $\boldsymbol{x}_3$, and $\hat{y}_4$ for $\boldsymbol{x}_4$. Please further compute the classification accuracy: $\frac{1}{4}\sum_{i=1}^{4} \mathbf{1}[\hat{y}_i == y_i]$. Do you see the importance of nonlinearity?