OSU CSE 3521/5521
Homework #4: Problem Set

Release Date: November 22th, 2020

Submission Instructions

Due Date: December 4st (23:59 ET), 2020

Submission: Please submit your solutions in a single PDF file named HW_4_name.number.pdf (e.g., HW_4_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. However, you need to write your own solutions and submit them separately. Also in your written report, you need to list with whom you have discussed for each problem. Please consult the syllabus for what is and is not acceptable collaboration.

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

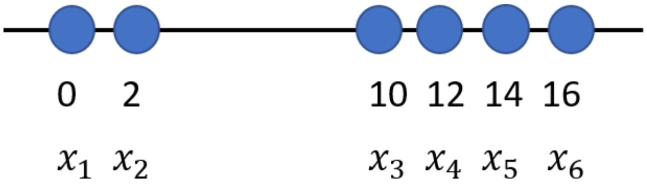


Figure 1: A one-dimensional data set of 6 data instances

1 Gaussian mixture models [11 points]

Figure 1 shows a data set of 6 data instances, each of them is one-dimensional. In the following, you will practice some steps of constructing a Gaussian mixture model with two components (i.e., 1 & 2) to model the distribution of 6 points.

Note that to construct a Gaussian mixture model, you are to estimate

- μ_1, σ_1, such that $p(X = x|Y = 1) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp - \frac{(x-\mu_1)^2}{2{\sigma_1}^2}$,
- μ_2, σ_2, such that $p(X = x|Y = 2) = \frac{1}{\sigma_2\sqrt{2\pi}} \exp - \frac{(x-\mu_2)^2}{2{\sigma_2}^2}$,
- λ, such that $p(Y = 1) = \lambda$,

so that together you have the GMM distribution

$$p(X = x) = p(Y = 1)p(X = x|Y = 1) + p(Y = 2)p(X = x|Y = 2).$$

In the following, let us denote by θ all the parameters $\{\mu_1, \sigma_1, \mu_2, \sigma_2, \lambda\}$.

Like K-means, GMM also needs initialization. Let

- $\mu_1^{(0)} = 4$, $\sigma_1^{(0)} = 4$
- $\mu_2^{(0)} = 10$, $\sigma_2^{(0)} = 4$
- $\lambda^{(0)} = 0.5$

be the initialization and $\theta^{(0)}$ as the notation to denote them all. $^{(0)}$ means the parameters at the 0-th iteration.

1. **E-steps [6 points]**: Please compute $p(Y = 1|x_i; \theta^{(0)})$ for each data point $\{x_i\}_{i=1}^6$ (a numerical value).

$$p(Y = 1|x_i; \theta^{(0)}) = \frac{p(Y = 1, x_i; \theta^{(0)})}{p(x_i; \theta^{(0)})}$$
$$= \frac{p(Y = 1; \theta^{(0)})p(x_i|Y = 1; \theta^{(0)})}{p(Y = 1; \theta^{(0)})p(x_i|Y = 1; \theta^{(0)}) + p(Y = 2; \theta^{(0)})p(x_i|Y = 2; \theta^{(0)})}$$

2. **M-steps [5 points]**: Now given $p(Y=1|x_i;\theta^{(0)})$ for $i \in \{1,\cdots,6\}$, let us compute the updated parameters $\theta^{(1)} = \{\mu_1^{(1)}, \sigma_1^{(1)}, \mu_2^{(1)}, \sigma_2^{(1)}, \lambda^{(1)}\}$.

$$\begin{aligned}
\mu_k^{(1)} &= \frac{\sum_i p(Y=k|x_i;\theta^{(0)}) \times x_i}{\sum_i p(Y=k|x_i;\theta^{(0)})}, \\
\sigma_k^{(1)} &= \sqrt{\frac{\sum_i p(Y=k|x_i;\theta^{(0)}) \times (x_i - \mu_k^{(1)})^2}{\sum_i p(Y=k|x_i;\theta^{(0)})}}, \\
\lambda^{(1)} &= \frac{\sum_i p(Y=1|x_i;\theta^{(0)})}{6},
\end{aligned}$$

where $k \in \{1,2\}$ is the index of each component. Please compute and write down the values of $\mu_1^{(1)}, \sigma_1^{(1)}, \mu_2^{(1)}, \sigma_2^{(1)}, \lambda^{(1)}$.

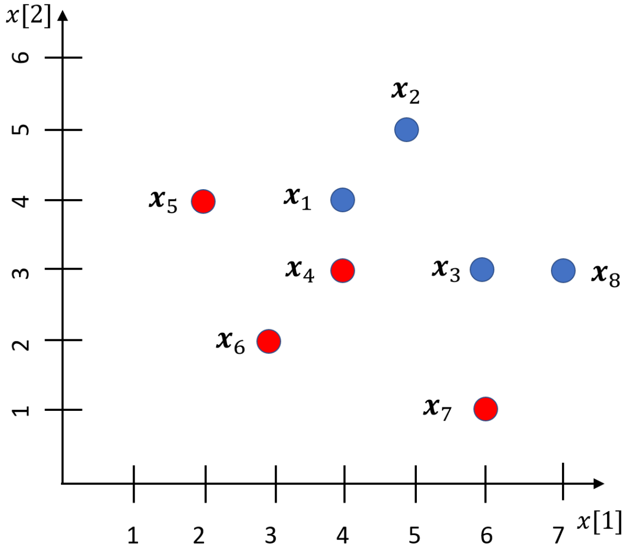


Figure 2: A two-dimensional, two-class data set of 8 data instances

2 Cross-validation for K nearest neighbor (KNN) classifiers [8 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. 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 2). 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

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

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

1. [**4 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. [**4 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 $D \setminus \{(x_j, y_j)\}$.

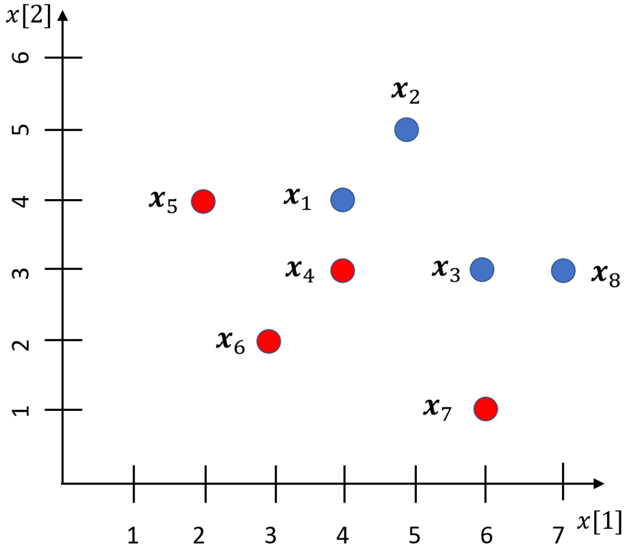


Figure 3: A two-dimensional, two-class data set of 8 data instances

3 Decision trees [14 points]

Figure 3 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 test 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 4. 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:

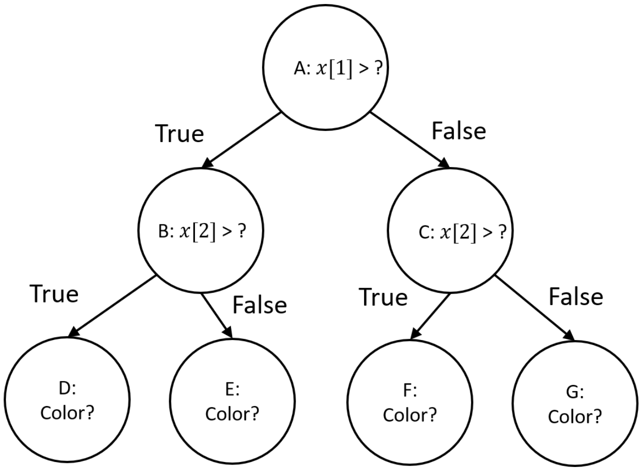


Figure 4: A decision tree of 7 nodes.

- 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.

4 Logistic regression [7 points]

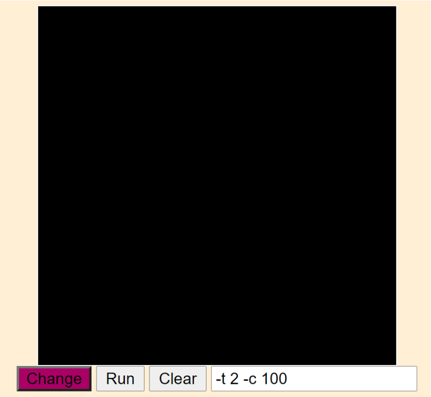
In the lectures, we talked about the loss function of logistic regression $\boldsymbol{w}^\top \boldsymbol{x} + b$ for a labeled data instance $(\boldsymbol{x}, y \in \{0, 1\})$ is

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

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. **[4 points]** $\frac{\partial \ell}{\partial w}$ at $w = 1$ and $b = 1$
2. **[3 points]** $\frac{\partial \ell}{\partial b}$ at $w = 1$ and $b = 1$



-t 2 -c 100

Figure 5: The LibSVM Graphic Interface.

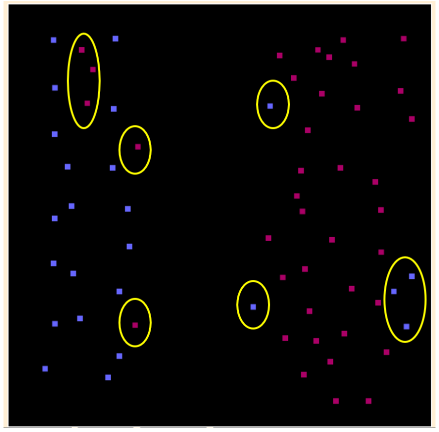


Figure 6: A two-class, two-dimensional data set. The yellow circles highlight the points that are a bit like outliers.

5 SVM [10 points]

Please go to `https://www.csie.ntu.edu.tw/~cjlin/libsvm/` and use the Graphic Interface (shown in Figure 5). Please construct a data set similar to Figure 6 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. [**1 points**] Which option will you use to construct a classifier for the future test data? and why (a short paragraph)?