Statistics 5525: Homework 1

Linjun Li, Department of Statistics, Virginia Tech

For each homework assignment, turn in at the beginning of class on the indicated due date. Late assignments will only be accepted with special permission. Write each problem up *very* neatly (LaTeX is preferred). Show all of your work.

Problem 1

From chapter 2 in TESL, the 2-dimensional dataset is created using the following procedure:

1. sample $m_i \sim N([1,0]', I)$ for $i = 1, \dots, 5$.
2. sample $m_i \sim N([0,1]', I)$ for $i = 6, \dots, 10$.
3. sample m from $(m_1, \dots, m_{10})$,
   each with probability 1/10 and sample $x_j \sim N(m, I/5)$.
4. if $m \in \{m_1, \dots, m_5\}$, let $x_j \in \{\text{class 1}\}$ (otherwise $x_j \in \{\text{class 2}\}$).
   (Note: save the subclass information (subclasses $1, \dots, 10$). You will use this later.)
5. repeat steps 3 and 4 for $j = 1, \dots, 100$.

Part a

Simulate from the above procedure (Gaussian mixture model), and plot the data. Use 2 separate colors to denote the two classes.

See Figure 1, the blue points belong to class 1, whereas the orange points belong to class 2. The red points are generated according to step 1 and step 2. The python code for this part can be found in Appendix I.

Part b

Use the least squares method to classify the data. Show a plot denoting your linear separating boundary. State both false positive and false negative rates.

The least squares method gives $\widehat{\beta} = (X'X)^{-1}X'Y$. Then we categorize points satisfying $X\widehat{\beta} < 0.5$ into class 1, and categorize points satisfying $X\widehat{\beta} > 0.5$ into class 2. The separating boundary can be found in Figure 1. The false positive rate by this classifier is 0.089286, whereas the false negative rate is 0.068182. The python code for this part can also be found in Appendix I.

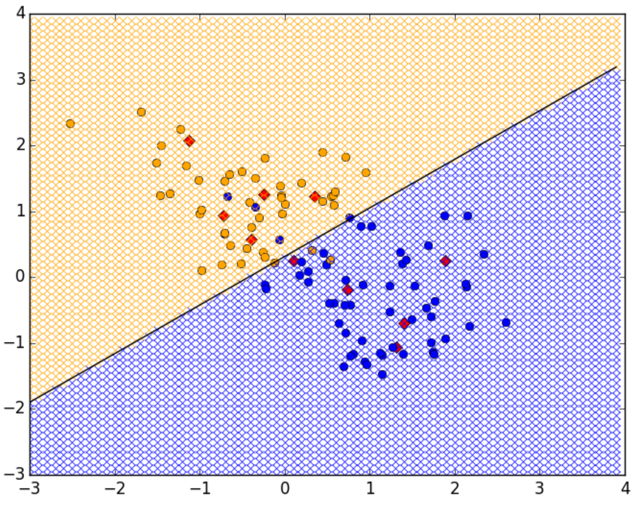

Figure 1: Simulated data points and the classification results by the least squares methods.

Problem 2

Given knowledge of all the parameters in the model ($\{m_1, \ldots m_{10}\}$), the mixture weights (1/10 for each subclass), and the covariance functions, derive the optimal separating boundary.

With the given information, we can build a Bayesian classifier. The optimal separating boundary can be derived as follows:

$$Pr(c_j|x, m_j) \propto P(x|m_j, c_j)Pr(c_j), Pr(c_j) = \frac{1}{10} \quad (1)$$

$$Pr(c_j|x, m_j) \propto P(x|m_j, c_j) \quad (2)$$

$$For super class 1 and super class 2, since the subclasses and mutually exclusive, we have: \quad (3)$$

$$Pr(C_1|x) = \sum_{j=1}^{5} Pr(c_j|x, m_j) \propto \sum_{j=1}^{5} P(x|m_j, c_j) \tag{4}$$

$$Pr(C_2|x) = \sum_{j=6}^{10} Pr(c_j|x, m_j) \propto \sum_{j=6}^{10} P(x|m_j, c_j) \tag{5}$$

To derive the separating boundary of superclass C_1 and C_2, we need to set $Pr(C_1|x) = Pr(C_2|x)$. Equivalently, we should set $\sum_{j=1}^{5} P(x|m_j, c_j) = \sum_{j=6}^{10} P(x|m_j, c_j)$. More explicitly, we need the following condition for setting up the separating boundary:

$$\sum_{j=1}^{5} e^{-\frac{5}{2}\|\vec{x}-\vec{m_j}\|^2} = \sum_{j=6}^{10} e^{-\frac{5}{2}\|\vec{x}-\vec{m_j}\|^2} \tag{6}$$

Thus, Eq.(6) gives optimal separating boundary.

Problem 3

Show a plot denoting this boundary. (Note: This may look a little different than what is in the book since your $\{m_1, \ldots m_{10}\}$ are different from what they've simulated). State both false positive and false negative rates.

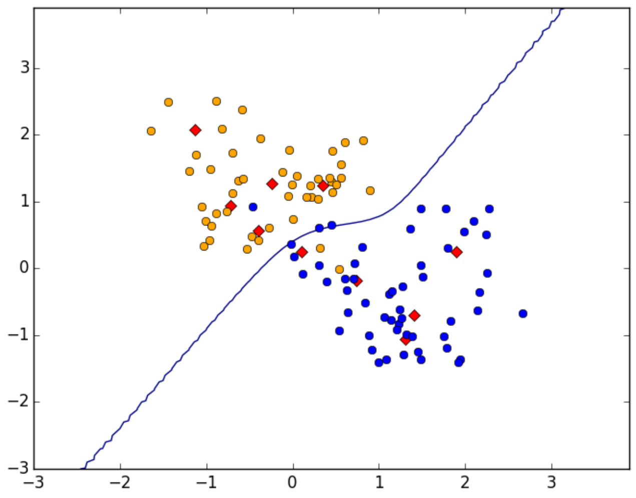

Figure 2: Bayesian classification and the corresponding separating boundary.

Figure 2 shows a plot of the result of Bayesian classification. The false positive rate is 0.056604, whereas the false negative rate is 0.042553. The python code for this part can also be found in Appendix II.

Problem 4

If you aren't given knowledge of the m_is, but are given the subclass labels, show how to construct the separating boundary. (Note: many methods exist, and some are better than others. You may use any method you deem reasonable). Show a plot of your results, and state both false positive and false negative rates.

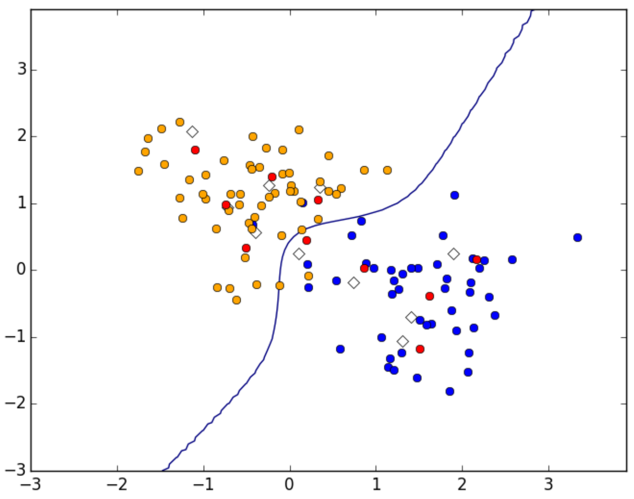

Figure 3: KNN classification with uniform weight and its separating boundary.

If the values of m_j's are not given, we can estimate the values of m_j's. One reasonable way to estimate m_j is to take the average of the locations of all data points that belong to the subclass C_j. That is, calculate $\widehat{m_j} = (\bar{x}_{c=c_j}, \bar{y}_{c=c_j})$, and plug $\widehat{m_j}$ back into the Eq.(6) for further calculation. Here in Figure 3 the white diamonds are true locations of the m_j's (not used for classification in Problem 4) and the red dots are the locations of the $\widehat{m_j}$. The false positive rate is 0.043478, the false negative rate is 0.018519.

Problem 5

If you weren't provided the $\{m_1, \dots m_{10}\}$ and subclass labels, but were given the superclass labels (class 1,2), what would you do? You don't have to derive anything, or code anything up. Simply describe in plain english.

With only superclass labels, we can use the K-Nearest Neighbor (KNN) method to do the classification. The implementation of KNN is as follows, we first create a fine mesh grid, then each point of the mesh grid is classified by a majority vote of its (k nearest) neighbors' superclass labels. After each point on the mesh grid is classified, we will have a good idea where the separating boundary is. A plot of the

KNN algorithm with uniform weight of votes can be found in Figure 4. A plot of the KNN algorithm with weight $1/d_i$ of vote can be found in Figure 5. (d_i is the distance from the point in question and its i^{th} neighbor, where $i = 1, 2, ..., k$.)

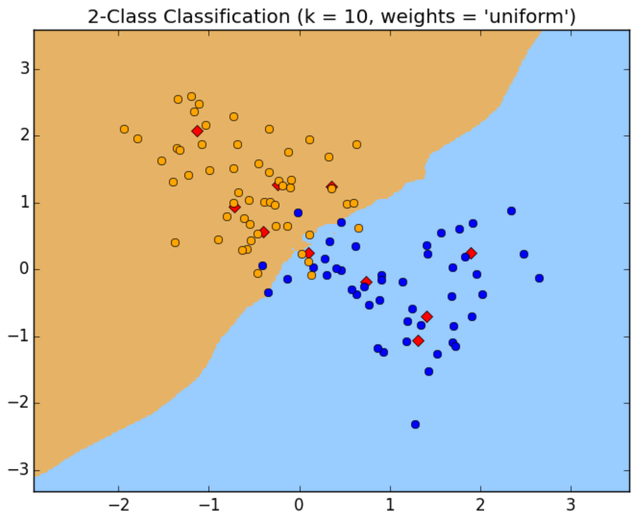

Figure 4: KNN classification with uniform weight and its separating boundary.

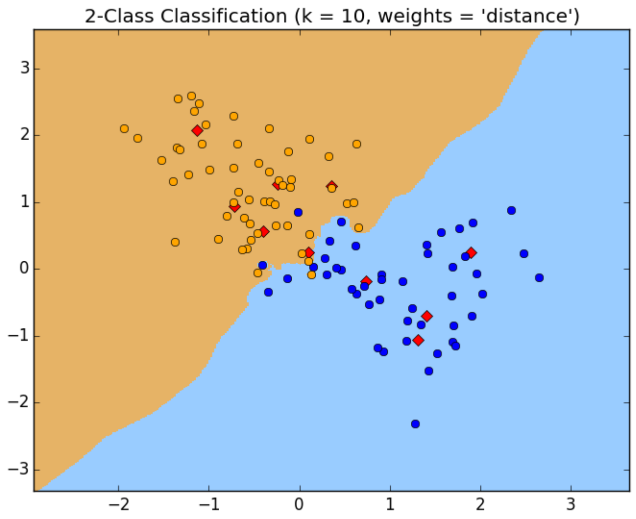

Figure 5: KNN classification with distance weight and its separating boundary.