

Aprendizagem 2023  
Homework II – Group 28

Gonalo Barias (ist1103124) & Raquel Braunschweig (ist1102624)

**Part I: Pen and Paper**

Consider the following dataset ( $y_3 - y_5$  are all categorical variables and the domain of  $y_2$  is  $[0, 1]$ ):

$D$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$
$\mathbf{x}_1$	0.24	0.36	1	1	0	A
$\mathbf{x}_2$	0.16	0.48	1	0	1	A
$\mathbf{x}_3$	0.32	0.72	0	1	2	A
$\mathbf{x}_4$	0.54	0.11	0	0	1	B
$\mathbf{x}_5$	0.66	0.39	0	0	0	B
$\mathbf{x}_6$	0.76	0.28	1	0	2	B
$\mathbf{x}_7$	0.41	0.53	0	1	1	B
$\mathbf{x}_8$	0.38	0.52	0	1	0	A
$\mathbf{x}_9$	0.42	0.59	0	1	1	B

1. Consider  $x_1 - x_7$  to be training observations,  $x_8 - x_9$  to be testing observations,  $y_1 - y_5$  to be input variables and  $y_6$  to be the target variable.

**Hint:** you can use `scipy.stats.multivariate_normal` for multivariate distribution calculus

- (a) **Learn a Bayesian classifier assuming:** i)  $\{y_1, y_2\}$ ,  $\{y_3, y_4\}$  and  $\{y_5\}$  sets of independent variables (e.g.,  $y_1 \perp\!\!\!\perp y_3$  yet  $y_1 \not\perp\!\!\!\perp y_2$ ), and ii)  $y_1 \times y_2 \in \mathbb{R}^2$  is normally distributed. Show all parameters (distributions and priors for subsequent testing).

As stated by the question prompt, variable sets  $\{y_1, y_2\}$ ,  $\{y_3, y_4\}$  and  $\{y_5\}$  are independent. Since we have seven training observations, we'll use those to train a Bayesian classifier.

We'll refer to the outcome, which can be A or B, as class.

To estimate  $P(\text{class}|y_1, y_2, y_3, y_4, y_5)$ , we can use Bayes' theorem:

$$P(\text{class}|y_1, y_2, y_3, y_4, y_5) = \frac{P(y_1, y_2, y_3, y_4, y_5|\text{class}) \times P(\text{class})}{P(y_1, y_2, y_3, y_4, y_5)} \quad (1)$$

Since we know  $\{y_1, y_2\}$ ,  $\{y_3, y_4\}$  and  $\{y_5\}$  are independent, we can rewrite  $P(y_1, y_2, y_3, y_4, y_5)$  as  $P(y_1, y_2) \cdot P(y_3, y_4) \cdot P(y_5)$ . Rewriting (1) with this, results in:

$$P(\text{class}|y_1, y_2, y_3, y_4, y_5) = \frac{P(y_1, y_2|\text{class})P(y_3, y_4|\text{class})P(y_5|\text{class}) \times P(\text{class})}{P(y_1, y_2)P(y_3, y_4)P(y_5)} \quad (2)$$

Given a new observation  $O$ , we are able to classify it by calculating  $P(\text{class}|O)$  for all classes and selecting the class with the highest probability as our prediction.

$$\begin{aligned}
\hat{c} &= \arg \max_{c \in \{A, B\}} \{P(c|O)\} \\
&= \arg \max_{c \in \{A, B\}} \left\{ \frac{P(y_1, y_2|c)P(y_3, y_4|c)P(y_5|c) \times P(c)}{P(y_1, y_2)P(y_3, y_4)P(y_5)} \right\} \\
&= \arg \max_{c \in \{A, B\}} \{P(y_1, y_2|c)P(y_3, y_4|c)P(y_5|c) \times P(c)\} \quad (\text{we can remove parameters that don't depend on } c)
\end{aligned} \tag{3}$$

We can therefore start calculating all these parameters.

**Note:** Even though  $P(y_1, y_2)$ ,  $P(y_3, y_4)$  and  $P(y_5)$  are not necessary to apply the model, we'll still calculate them for the sake of showing all parameters.

Calculating  $P(A)$ ,  $P(B)$  and all parameters involving  $y_1$  through  $y_5$  is straightforward, since they can be inferred from the table.

We have 3 observations of A and 4 observations of B, out of a total of 7 training observations. Therefore,

$$P(A) = \frac{3}{7} \quad P(B) = \frac{4}{7}$$

In a similar manner we can obtain the probabilities for  $y_5$ ,

$$P(y_5 = 0) = \frac{2}{7} \quad P(y_5 = 1) = \frac{3}{7} \quad P(y_5 = 2) = \frac{2}{7}$$

Now for the conditional probabilities of  $y_5$ ,

$$\begin{aligned}
P(y_5 = 0 | A) &= \frac{1}{3}, \quad P(y_5 = 1 | A) = \frac{1}{3}, \quad P(y_5 = 2 | A) = \frac{1}{3} \\
P(y_5 = 0 | B) &= \frac{1}{4}, \quad P(y_5 = 1 | B) = \frac{2}{4}, \quad P(y_5 = 2 | B) = \frac{1}{4}
\end{aligned}$$

For the four possible combinations of  $y_3$  and  $y_4$  we can follow the same logic as above,

$$\begin{aligned}
P(y_3 = 0, y_4 = 0) &= \frac{2}{7}, \quad P(y_3 = 0, y_4 = 1) = \frac{2}{7} \\
P(y_3 = 1, y_4 = 0) &= \frac{2}{7}, \quad P(y_3 = 1, y_4 = 1) = \frac{1}{7}
\end{aligned}$$

Finally, considering each class and the four possible combinations of  $y_3$  and  $y_4$ , we can use the table to calculate the following:

$$\begin{aligned}
P(y_3 = 0, y_4 = 0 | A) &= \frac{0}{3}, \quad P(y_3 = 0, y_4 = 1 | A) = \frac{1}{3} \\
P(y_3 = 1, y_4 = 0 | A) &= \frac{1}{3}, \quad P(y_3 = 1, y_4 = 1 | A) = \frac{1}{3} \\
P(y_3 = 0, y_4 = 0 | B) &= \frac{2}{4}, \quad P(y_3 = 0, y_4 = 1 | B) = \frac{1}{4} \\
P(y_3 = 1, y_4 = 0 | B) &= \frac{1}{4}, \quad P(y_3 = 1, y_4 = 1 | B) = \frac{0}{4}
\end{aligned}$$

Calculating now the parameters related to the variable set  $\{y_1, y_2\}$ . We know that  $(y_1, y_2)$  follows a Multivariate Gaussian Distribution. Therefore,

$$P((y_1, y_2) | \mu, \Sigma) = \mathcal{N}((y_1, y_2) | \mu, \Sigma) \tag{4}$$

We can use the observations we have to approximate a value for the 2-dimensional mean vector ( $\mu$ ) and the covariance matrix ( $\Sigma$ ).

$$\begin{aligned}\mu &= \frac{1}{7} \sum_{i=1}^7 \begin{bmatrix} y_{1,i} \\ y_{2,i} \end{bmatrix} = \frac{1}{7} \left( \begin{bmatrix} 0.24 \\ 0.36 \end{bmatrix} + \begin{bmatrix} 0.16 \\ 0.48 \end{bmatrix} + \begin{bmatrix} 0.32 \\ 0.72 \end{bmatrix} + \begin{bmatrix} 0.54 \\ 0.11 \end{bmatrix} + \begin{bmatrix} 0.66 \\ 0.39 \end{bmatrix} + \begin{bmatrix} 0.76 \\ 0.28 \end{bmatrix} + \begin{bmatrix} 0.41 \\ 0.53 \end{bmatrix} \right) \\ &= \begin{bmatrix} 0.4414 \\ 0.41 \end{bmatrix}\end{aligned}$$

$$\Sigma_{00} = \frac{1}{N-1} \sum_{i=1}^N (y_{1,i} - \mu_1)^2 = \frac{1}{7-1} [(0.24 - 0.4414)^2 + \dots + (0.41 - 0.4414)^2] \approx 0.0491$$

$$\Sigma_{11} = \frac{1}{N-1} \sum_{i=1}^N (y_{2,i} - \mu_2)^2 = \frac{1}{7-1} [(0.36 - 0.41)^2 + \dots + (0.53 - 0.41)^2] \approx 0.0375$$

$$\begin{aligned}\Sigma_{01} = \Sigma_{10} &= \frac{1}{N-1} \sum_{i=1}^N (y_{1,i} - \mu_1)(y_{2,i} - \mu_2) \\ &= \frac{1}{7-1} [(0.24 - 0.4414)(0.36 - 0.41) + \dots + (0.41 - 0.4414)(0.53 - 0.41)] \\ &\approx -0.0211\end{aligned}$$

$$\Sigma = \begin{bmatrix} \Sigma_{00} & \Sigma_{10} \\ \Sigma_{01} & \Sigma_{11} \end{bmatrix} = \begin{bmatrix} 0.0491 & -0.0211 \\ -0.0211 & 0.0375 \end{bmatrix}$$

Therefore,  $P(y_1, y_2) \sim \mathcal{N}\left((y_1, y_2) | \mu = \begin{bmatrix} 0.4414 \\ 0.41 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.0491 & -0.0211 \\ -0.0211 & 0.0375 \end{bmatrix}\right)$ .

We can repeat the process for both classes (A and B).

Starting with A:

$$\mu = \frac{1}{3} \sum_{i=1}^3 \begin{bmatrix} y_{1,i}|A \\ y_{2,i}|A \end{bmatrix} = \frac{1}{3} \left( \begin{bmatrix} 0.24 \\ 0.36 \end{bmatrix} + \begin{bmatrix} 0.16 \\ 0.48 \end{bmatrix} + \begin{bmatrix} 0.32 \\ 0.72 \end{bmatrix} \right) = \begin{bmatrix} 0.24 \\ 0.52 \end{bmatrix}$$

$$\Sigma_{00} = \frac{1}{N-1} \sum_{i=1}^N (y_{1,i}|A - \mu_1)^2 = \frac{1}{3-1} [(0.24 - 0.24)^2 + \dots + (0.32 - 0.24)^2] \approx 0.0064$$

$$\Sigma_{11} = \frac{1}{N-1} \sum_{i=1}^N (y_{2,i}|A - \mu_2)^2 = \frac{1}{3-1} [(0.36 - 0.52)^2 + \dots + (0.72 - 0.52)^2] \approx 0.0336$$

$$\begin{aligned}\Sigma_{01} = \Sigma_{10} &= \frac{1}{N-1} \sum_{i=1}^N (y_{1,i}|A - \mu_1)(y_{2,i}|A - \mu_2) \\ &= \frac{1}{3-1} [(0.24 - 0.24)(0.36 - 0.52) + \dots + (0.32 - 0.24)(0.72 - 0.52)] \approx 0.0096\end{aligned}$$

$$\Sigma = \begin{bmatrix} \Sigma_{00} & \Sigma_{10} \\ \Sigma_{01} & \Sigma_{11} \end{bmatrix} = \begin{bmatrix} 0.0064 & 0.0096 \\ 0.0096 & 0.0336 \end{bmatrix}$$

Therefore,  $P(y_1, y_2|A) \sim \mathcal{N}\left((y_1, y_2) | \mu = \begin{bmatrix} 0.24 \\ 0.52 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.0064 & 0.0096 \\ 0.0096 & 0.0336 \end{bmatrix}\right)$ .

And now the B:

$$\begin{aligned}\boldsymbol{\mu} &= \frac{1}{4} \sum_{i=1}^4 \begin{bmatrix} y_{1,i} | \mathbf{B} \\ y_{2,i} | \mathbf{B} \end{bmatrix} = \frac{1}{4} \left( \begin{bmatrix} 0.54 \\ 0.11 \end{bmatrix} + \begin{bmatrix} 0.66 \\ 0.39 \end{bmatrix} + \begin{bmatrix} 0.76 \\ 0.28 \end{bmatrix} + \begin{bmatrix} 0.41 \\ 0.53 \end{bmatrix} \right) = \begin{bmatrix} 0.5925 \\ 0.3275 \end{bmatrix} \\ \Sigma_{00} &= \frac{1}{N-1} \sum_{i=1}^N (y_{1,i} | \mathbf{B} - \mu_1)^2 = \frac{1}{4-1} [(0.54 - 0.5925)^2 + \dots + (0.41 - 0.5925)^2] \approx 0.0229 \\ \Sigma_{11} &= \frac{1}{N-1} \sum_{i=1}^N (y_{2,i} | \mathbf{B} - \mu_2)^2 = \frac{1}{4-1} [(0.11 - 0.3275)^2 + \dots + (0.53 - 0.3275)^2] \approx 0.0315 \\ \Sigma_{01} &= \Sigma_{10} = \frac{1}{N-1} \sum_{i=1}^N (y_{1,i} | \mathbf{B} - \mu_1)(y_{2,i} | \mathbf{B} - \mu_2) \\ &= \frac{1}{4-1} [(0.54 - 0.5925)(0.11 - 0.3275) + \dots + (0.41 - 0.5925)(0.53 - 0.3275)] \approx -0.0098 \\ \Sigma &= \begin{bmatrix} \Sigma_{00} & \Sigma_{10} \\ \Sigma_{01} & \Sigma_{11} \end{bmatrix} = \begin{bmatrix} 0.0229 & -0.0098 \\ -0.0098 & 0.0315 \end{bmatrix}\end{aligned}$$

Therefore,  $P(y_1, y_2 | \mathbf{B}) \sim \mathcal{N}\left((y_1, y_2) | \boldsymbol{\mu} = \begin{bmatrix} 0.5925 \\ 0.3275 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.0229 & -0.0098 \\ -0.0098 & 0.0315 \end{bmatrix}\right)$ .

We now have all the parameters necessary to apply the Bayesian classifier to new observations.

(b) **Under a MAP assumption, classify each testing observation showing all of your calculus.**

Since we will need the  $(y_1, y_2)$  pdf values for the following exercises, let's calculate them using `scipy.multivariate_normal`, just like it was suggested in the hint:

```
1 from scipy.stats import multivariate_normal
2
3 x8 = multivariate_normal.pdf([0.38, 0.52], mean=[0.4414, 0.41],
4                               cov=[[0.0491, -0.0211], [-0.0211, 0.0375]])
5 x9 = multivariate_normal.pdf([0.42, 0.59], mean=[0.4414, 0.41],
6                               cov=[[0.0491, -0.0211], [-0.0211, 0.0375]])
7
8 x8_A = multivariate_normal.pdf([0.38, 0.52], mean=[0.24, 0.52],
9                               cov=[[0.0064, 0.0096], [0.0096, 0.0336]])
10 x8_B = multivariate_normal.pdf([0.38, 0.52], mean=[0.5925, 0.3275],
11                                cov=[[0.0229, -0.0098], [-0.0098, 0.03149]])
12 x9_A = multivariate_normal.pdf([0.42, 0.59], mean=[0.24, 0.52],
13                                cov=[[0.0064, 0.0096], [0.0096, 0.0336]])
14 x9_B = multivariate_normal.pdf([0.42, 0.59], mean=[0.5925, 0.3275],
15                                cov=[[0.0229, -0.0098], [-0.0098, 0.03149]])
```

From the code above we get the following:

$$P(y_1 = 0.38, y_2 = 0.52) \approx 3.6225, \quad P(y_1 = 0.42, y_2 = 0.59) \approx 2.5387$$

$$P(y_1 = 0.38, y_2 = 0.52 | A) \approx 0.9847, \quad P(y_1 = 0.38, y_2 = 0.52 | B) \approx 1.9623$$

$$P(y_1 = 0.42, y_2 = 0.59 | A) \approx 0.4031, \quad P(y_1 = 0.42, y_2 = 0.59 | B) \approx 1.7285$$

Since we are under a Maximum a Posteriori assumption, we can just use the expression at (3) and replace the values for each of the testing observations ( $x_8$  and  $x_9$ ).

Starting with  $x_8$ :

$$P(y_1 = 0.38, y_2 = 0.52|A)P(y_3 = 0, y_4 = 1|A)P(y_5 = 0|A) \times P(A) = 0.9847 \times \frac{1}{3} \times \frac{1}{3} \times \frac{3}{7} \approx 0.04689$$

$$P(y_1 = 0.38, y_2 = 0.52|B)P(y_3 = 0, y_4 = 1|B)P(y_5 = 0|B) \times P(B) = 1.9623 \times \frac{1}{4} \times \frac{1}{4} \times \frac{4}{7} \approx 0.07008$$

$$\begin{aligned}\hat{z}_{x_8} &= \arg \max_{c \in \{A, B\}} \{P(y_1 = 0.38, y_2 = 0.52|c)P(y_3 = 0, y_4 = 1|c)P(y_5 = 0|c) \times P(c)\} \\ &= \arg \max \{P(y_1, y_2|A)P(y_3, y_4|A)P(y_5|A) \times P(A); P(y_1, y_2|B)P(y_3, y_4|B)P(y_5|B) \times P(B)\} \\ &= B\end{aligned}$$

And now with  $x_9$ :

$$P(y_1 = 0.42, y_2 = 0.59|A)P(y_3 = 0, y_4 = 1|A)P(y_5 = 1|A) \times P(A) = 0.4031 \times \frac{1}{3} \times \frac{1}{3} \times \frac{3}{7} \approx 0.0192$$

$$P(y_1 = 0.42, y_2 = 0.59|B)P(y_3 = 0, y_4 = 1|B)P(y_5 = 1|B) \times P(B) = 1.7285 \times \frac{1}{4} \times \frac{2}{4} \times \frac{4}{7} \approx 0.1235$$

$$\begin{aligned}\hat{z}_{x_9} &= \arg \max_{c \in \{A, B\}} \{P(y_1 = 0.42, y_2 = 0.59|c)P(y_3 = 0, y_4 = 1|c)P(y_5 = 1|c) \times P(c)\} \\ &= \arg \max \{P(y_1, y_2|A)P(y_3, y_4|A)P(y_5|A) \times P(A); P(y_1, y_2|B)P(y_3, y_4|B)P(y_5|B) \times P(B)\} \\ &= B\end{aligned}$$

**Therefore**, we conclude that under a MAP assumption, observations  $x_8$  and  $x_9$  will be classified with B and B, respectively.

(c) **Consider that the default decision threshold of  $\theta = 0.5$  can be adjusted according to**

$$f(\mathbf{x}|\theta) = \begin{cases} A, & P(A|\mathbf{x}) > \theta \\ B, & \text{otherwise} \end{cases}$$

**Under a maximum likelihood assumption, what thresholds optimize testing accuracy?**

On question 1.b) we calculated the following values:

$$P(A) = \frac{3}{7}$$

$$P(B) = \frac{4}{7}$$

$$P(x_8|A) * P(A) \approx 0.04689$$

$$P(x_8|B) * P(B) \approx 0.07008$$

$$P(x_9|A) * P(A) \approx 0.0192$$

$$P(x_9|B) * P(B) \approx 0.1235$$

So, to discover the maximum likelihood ( $P(x_x|class)$ ), we have to split them by  $P(class)$ . Here are the results:

$$P(x_8|A) = 0.10941$$

$$P(x_8|B) = 0.12264$$

$$P(x_9|A) = 0.0448$$

$$P(x_9|B) \approx 0.2161$$

According to the *FAQ*, we can use the values we calculated as a **rough proxy for posteriors**. The next step is **normalize** them so we can get the maximum a posteriori:

$$P(A|x_8) = \frac{0.10941}{0.10941 + 0.12264} \approx 0.4715$$

$$P(B|x_8) = \frac{0.12264}{0.10941 + 0.12264} \approx 0.5285$$

$$P(A|x_9) = \frac{0.0448}{0.0448 + 0.2161} \approx 0.1717$$

$$P(B|x_9) = \frac{0.216125}{0.0448 + 0.2161} \approx 0.8283$$

**Therefore**, we can conclude that the threshold of values for  $\theta$  is  $[0.1717, 0.4715]$ .

2. Let  $y_1$  be the target numeric variable,  $y_2 - y_6$  be the input variables where  $y_2$  is binarized under an equal-width (equal-range) discretization. For the evaluation of regressors, consider a 3-fold cross-validation over the full dataset ( $x_1 - x_9$ ) without shuffling the observations.

- (a) **Identify the observations and features per data fold after the binarization procedure.**

To do the **binarization procedure** with an **equal-width discretization**, we need to divide  $y_2$  into two intervals. Which are:

$$\text{interval}_1 = [0, 0.5]$$

$$\text{interval}_2 = [0.5, 1]$$

**Here is the binarization of  $y_2$  based on those intervals:**

$D$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$
$x_1$	0.24	0	1	1	0	A
$x_2$	0.16	0	1	0	1	A
$x_3$	0.32	1	0	1	2	A
$x_4$	0.54	0	0	0	1	B
$x_5$	0.66	0	0	0	0	B
$x_6$	0.76	0	1	0	2	B
$x_7$	0.41	1	0	1	1	B
$x_8$	0.38	1	0	1	0	A
$x_9$	0.42	1	0	1	1	B

The next step is identifying our folds, which will be:

Fold 1 =  $x_1 x_2 x_3$

Fold 2 =  $x_4 x_5 x_6$

Fold 3 =  $x_7 x_8 x_9$

So our datasets will be:

Fold 1						
$D$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$
$x_1$	0.24	0	1	1	0	A
$x_2$	0.16	0	1	0	1	A
$x_3$	0.32	1	0	1	2	A

Fold 2						
$D$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$
$x_4$	0.54	0	0	0	1	B
$x_5$	0.66	0	0	0	0	B
$x_6$	0.76	0	1	0	2	B

Fold 3						
$D$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$
$x_7$	0.41	1	0	1	1	B
$x_8$	0.38	1	0	1	0	A
$x_9$	0.42	1	0	1	1	B

- (b) Consider a distance-weighted  $k$ NN with  $k = 3$ , Hamming distance ( $d$ ), and  $1 / d$  weighting. Compute the MAE of this  $k$ NN regressor for the 1<sup>st</sup> iteration of the cross-validation (i.e. train observations have the lower indices).

The formula for **weighted average**, considering that  $k=3$ , is the following:

$$\text{Weighted Average} = \frac{\frac{1}{d_1} \cdot y_1 + \frac{1}{d_2} \cdot y_2 + \frac{1}{d_3} \cdot y_3}{\frac{1}{d_1} + \frac{1}{d_2} + \frac{1}{d_3}} \quad (5)$$

And the equation for the **mean absolute error** is given by:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (6)$$

As stated in the prompt, we will use Folds 1 and 2 for training, reserving Fold 3 for testing. Let's start by computing the **Hamming distances**, for  $x_7$  first:

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
$H(x_7, x_j)$	4	4	2	2	3	4

**Now**, for  $x_8$ :

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
$H(x_8, x_j)$	2	4	1	4	3	5

**Finally**, for  $x_9$ :

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
$H(x_9, x_j)$	4	4	2	2	3	4

Now let's determine the **three closest neighbors** to each test observation:

- For  $x_7$  it is  $x_3$ ,  $x_4$  and  $x_5$
- For  $x_8$  it is  $x_1$ ,  $x_3$  and  $x_5$
- For  $x_9$  it is  $x_3$ ,  $x_4$  and  $x_5$

The next step is calculating their **weighted average**. By replacing the formula on (5), we get the following values:

Weighted average for  $x_7 = 0.4875$

Weighted average for  $x_8 = 0.36$

Weighted average for  $x_9 = 0.4875$

**Finally**, let's compute the mean absolute error by using the equation on (6):

$$MAE = 0.055$$

## Part II: Programming and critical analysis

Considering the `column_diagnosis.arff` dataset available at the course webpage's homework tab. Using `sklearn`, apply a 10-fold stratified cross-validation with shuffling (`random_state=0`) for the assessment of predictive models along this section.

1. **Compare the performance of  $k$ NN with  $k = 5$  and Naïve Bayes with Gaussian assumption (consider all remaining parameters for each classifier as `sklearn`'s default):**

(a) **Plot two boxplots with the fold accuracies for each classifier.**

```
1 import matplotlib.pyplot as plt, pandas as pd
2 from sklearn.model_selection import StratifiedKFold, cross_val_score
3 from sklearn.neighbors import KNeighborsClassifier
4 from sklearn.naive_bayes import GaussianNB
5 from scipy.io.arff import loadarff
6
7 # Read the ARFF file and prepare data
8 data = loadarff("./data/column_diagnosis.arff")
```



```

9 df = pd.DataFrame(data[0])
10 df["class"] = df["class"].str.decode("utf-8")
11 X, y = df.drop("class", axis=1), df["class"]
12
13 # Define cross-validation strategy
14 folds = StratifiedKFold(n_splits=10, shuffle=True, random_state=0)
15
16 # Initialize classifiers
17 knn_predictor = KNeighborsClassifier(n_neighbors=5)
18 nb_predictor = GaussianNB()
19
20 # Evaluate classifiers
21 knn_accs = cross_val_score(knn_predictor, X, y, cv=folds, scoring="accuracy")
22 nb_accs = cross_val_score(nb_predictor, X, y, cv=folds, scoring="accuracy")
23
24 # Plot boxplots
25 plt.figure(figsize=(7, 5))
26 b_plot = plt.boxplot(
27     [knn_accs, nb_accs], patch_artist=True, labels=["kNN", "Naive Bayes"]
28 )
29
30 colors = ["#f8766d", "#00bfc4"]
31 for patch, color in zip(b_plot["boxes"], colors):
32     patch.set_facecolor(color)
33 for median in b_plot["medians"]:
34     median.set_color("black")
35
36 plt.ylabel("Accuracy")
37 plt.grid(axis="y")
38 plt.show()

```

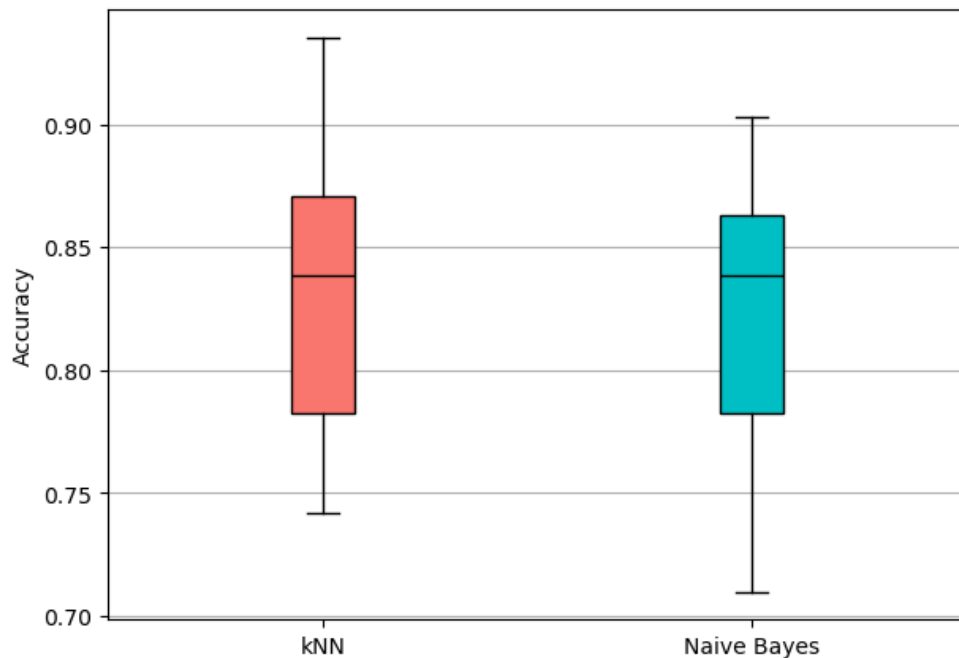


Figure 1: Boxplots with the fold accuracies of  $k$ NN ( $k = 5$ ) and Naïve Bayes

- (b) Using `scipy`, test the hypothesis "*k*NN is statistically superior to Naïve Bayes regarding accuracy", asserting whether is true.

We'll consider the null hypothesis and alternate hypothesis below and perform a single-tailed test using the accuracies obtained in the previous answer,

$$H_0 : \text{accuracy}_{k\text{NN}} = \text{accuracy}_{\text{Naïve Bayes}}$$

$$H_1 : \text{accuracy}_{k\text{NN}} > \text{accuracy}_{\text{Naïve Bayes}}$$

```
1 from scipy.stats import ttest_rel
2
3 # Is knn better than naive bayes?
4 res = ttest_rel(knn_accs, nb_accs, alternative="greater")
5 print("Is knn > naive bayes? pval =", res.pvalue)
```

Using `scipy` we get a p-value of, approximately,  $0.190428 = 19.0428\%$ .

This means we cannot reject the hypothesis  $H_0$  at common significance levels (1%, 5% and 10%).

**Therefore**, we cannot assert that *k*NN is statistically superior to Naïve Bayes. We also cannot state that the hypothesis on the statement is outright false without checking other statistical tests.

2. Consider two *k*NN predictors with  $k = 1$  and  $k = 5$  (uniform weights, Euclidean distance, all remaining parameters as default). Plot the differences between the two cumulative confusion matrices of the predictors. Comment.

```
1 import numpy as np, matplotlib.pyplot as plt, pandas as pd, seaborn as sns
2 from sklearn.model_selection import StratifiedKFold
3 from sklearn.neighbors import KNeighborsClassifier
4 from sklearn.metrics import confusion_matrix
5 from scipy.io.arff import loadarff
6
7 # Read the ARFF file and prepare data
8 data = loadarff("./data/column_diagnosis.arff")
9 df = pd.DataFrame(data[0])
10 df["class"] = df["class"].str.decode("utf-8")
11 X, y = df.drop("class", axis=1), df["class"]
12
13 # Initialize StratifiedKFold with 10 folds and shuffling
14 folds = StratifiedKFold(n_splits=10, shuffle=True, random_state=0)
15
16 # Create kNN classifiers with k=1 and k=5
17 knn_1 = KNeighborsClassifier(n_neighbors=1)
18 knn_5 = KNeighborsClassifier(n_neighbors=5)
19
20 labels = ["Hernia", "Normal", "Spondylolisthesis"]
21 cm_1, cm_5 = np.zeros((3, 3)), np.zeros((3, 3))
22 for train_k, test_k in folds.split(X, y):
23     X_train, X_test = X.iloc[train_k], X.iloc[test_k]
24     y_train, y_test = y.iloc[train_k], y.iloc[test_k]
25
26 # Fit kNN classifiers and assess
27 knn_1.fit(X_train, y_train)
28 knn_5.fit(X_train, y_train)
29 knn_1_pred, knn_5_pred = knn_1.predict(X_test), knn_5.predict(X_test)
```

```

30 cm_1 += np.array(confusion_matrix(y_test, knn_1_pred, labels=labels))
31 cm_5 += np.array(confusion_matrix(y_test, knn_5_pred, labels=labels))
32
33 # Calculate cumulative confusion matrices
34 cm_diff = cm_1 - cm_5
35 cm_diff_df = pd.DataFrame(cm_diff, index=labels, columns=labels)
36
37 # Plot the differences
38 plt.figure(figsize=(9, 7))
39 sns.heatmap(
40     cm_diff_df, cmap="Purples", annot=True, annot_kws={"fontsize": 14}, fmt="g"
41 )
42 plt.xlabel("Predicted")
43 plt.ylabel("Real")
44 plt.show()

```

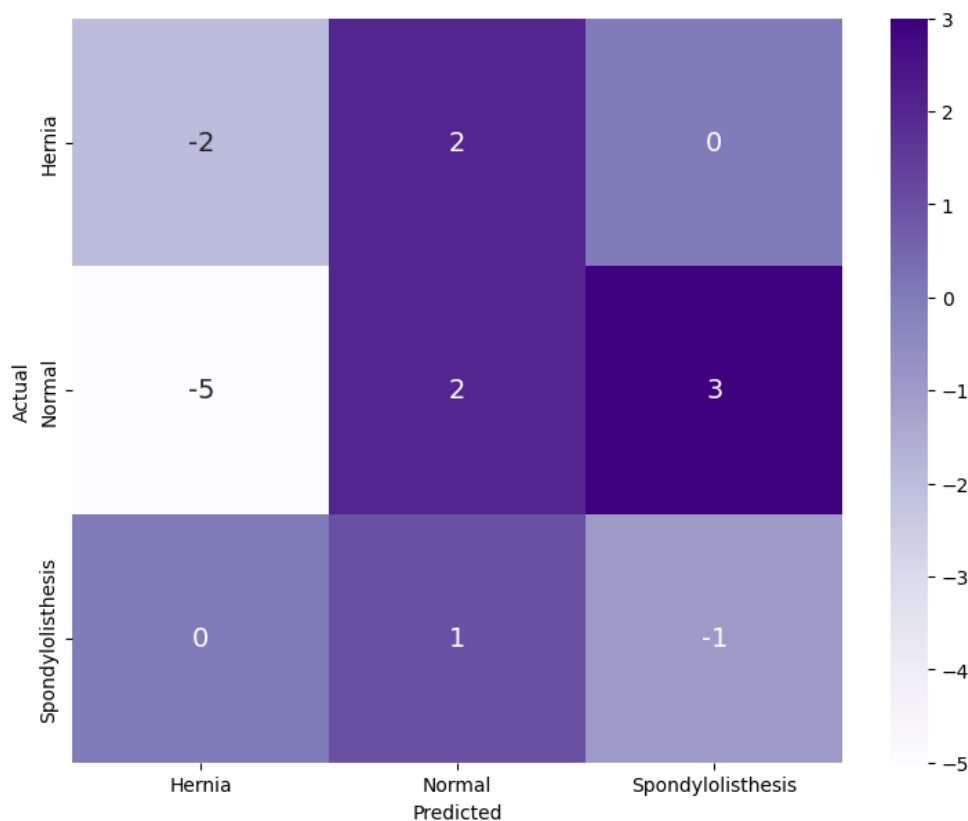


Figure 2: Confusion Matrix Differences Between k=1 and k=5 k-Nearest Neighbors ( $k$ NN) Classifiers

Blah

3. Considering the unique properties of `column_diagnosis`, identify three possible difficulties of Naïve Bayes when learning from the given dataset.

Here are three possible difficulties of Naïve Bayes when learning from the given dataset, in no particular order:

- Variable dependencies (inadequacy of independence assumption).

- Variables not normally distributed (inadequacy of Gaussian assumption). Probability estimates from a limited number of observations (e.g., inadequate estimates, null probabilities).
- Imbalanced class creating biases in MAP estimates via priors.