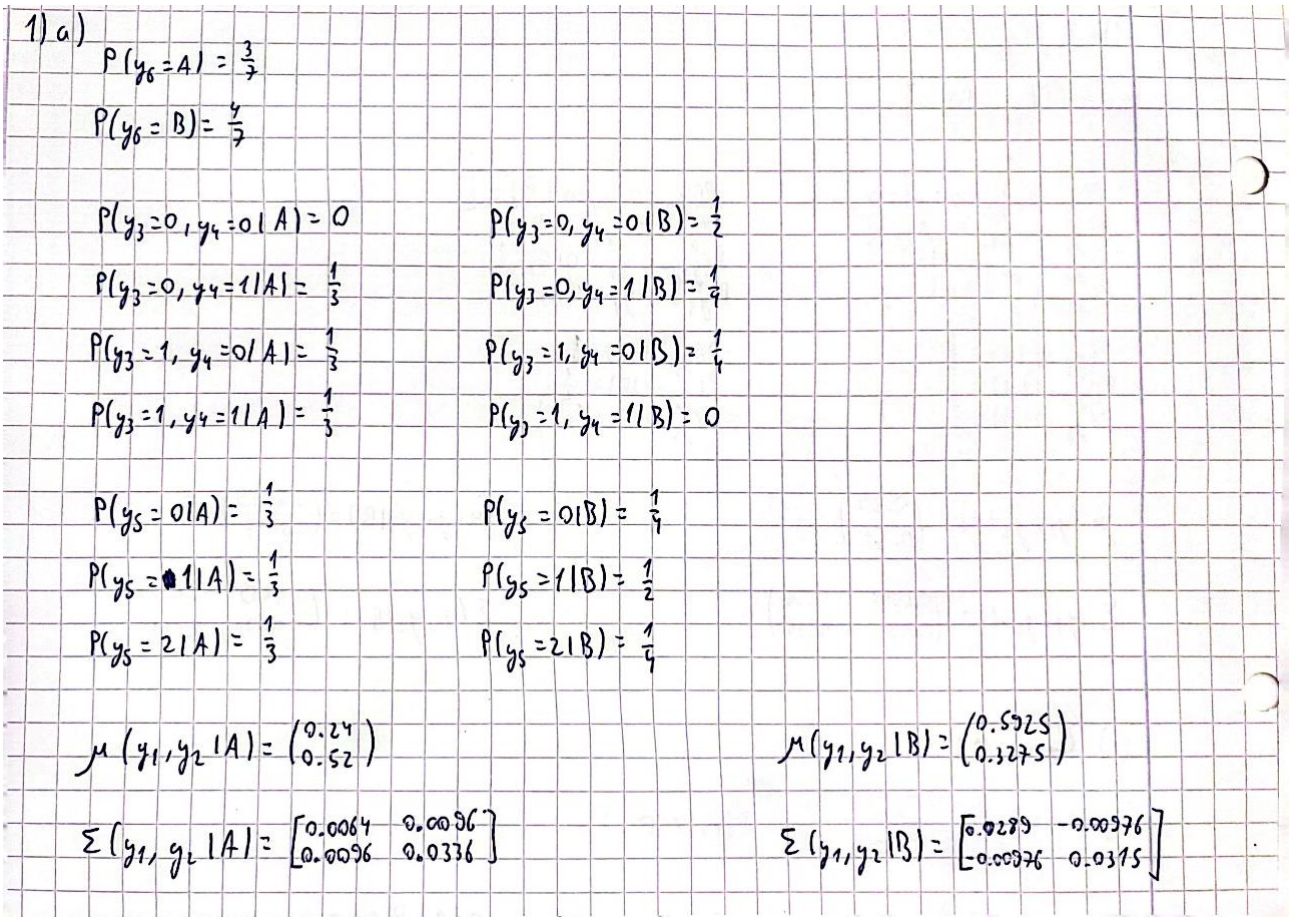
Aprendizagem 2022/23

Homework II – Group 107

(ist1103811, ist1103479)

I. Pen-and-paper

1)a)



Aprendizagem 2022/23

Homework II – Group 107

(ist1103811, ist1103479)

1)b)

b) Com MAP

$$P(y_6 = A \mid x_8) = \frac{P(x_8 \mid A)P(A)}{P(x_8)} \overset{MAP}{=} P(x_8 \mid A)P(A) =$$

$$= P(y_1 = 0.38, y_2 = 0.52, y_3 = 0, y_4 = 1, y_5 = 0 \mid A)\, P(A) =$$

$$= P(y_1 = 0.38, y_2 = 0.52 \mid A)\, P(y_3 = 0, y_4 = 1 \mid A)\, P(y_5 = 0 \mid A)\, P(A) =$$

$$= 0.422 \times \frac{1}{3} \times \frac{1}{3} \times \frac{3}{7} \simeq 0.0201$$

$$P(B \mid x_8) \overset{MAP}{=} \frac{P(x_8 \mid B)P(B)}{P(x_8)} = P(y_1 = 0.38, y_2 = 0.52 \mid B)\, P(y_3 = 0, y_4 = 1 \mid B)\, P(y_5 = 0 \mid B)\, P(B) =$$

$$= 1.121 \times \frac{1}{4} \times \frac{1}{4} \times \frac{4}{7} \simeq 0.04$$

Normalização: $P(A \mid x_8)_{normalizado} = \frac{P(A \mid x_8)}{P(A \mid x_8) + P(B \mid x_8)} = \frac{0.0201}{0.0201 + 0.04} \simeq 0.3344$

$$P(B \mid x_8)_{normalizado} = 1 - P(A \mid x_8)_{norm.} = 0.6656$$

Classificamos como (B)

$$P(A \mid x_9) \overset{MAP}{=} \frac{P(x_9 \mid A)P(A)}{P(x_9)} = P(y_1 = 0.42, y_2 = 0.59 \mid A)\, P(y_3 = 0, y_4 = 1 \mid A)\, P(y_5 = 1 \mid A)\, P(A) =$$

$$= 0.1727 \times \frac{1}{3} \times \frac{1}{3} \times \frac{3}{7} \simeq 0.00822$$

$$P(B \mid x_9) \overset{MAP}{=} \frac{P(x_9 \mid B)P(B)}{P(x_9)} = P(y_1 = 0.42, y_2 = 0.59 \mid B)\, P(y_3 = 0, y_4 = 1 \mid B)\, P(y_5 = 1 \mid B)\, P(B) =$$

$$= 0.9878 \times \frac{1}{4} \times \frac{1}{2} \times \frac{4}{7} \simeq 0.0706$$

Normalização: $P(A \mid x_9)_{norm.} = \frac{P(A \mid x_9)}{P(A \mid x_9) + P(B \mid x_9)} = \frac{0.00822}{0.00822 + 0.0706} \simeq 0.1043$

$$P(B \mid x_9)_{norm} = 1 - P(A \mid x_9)_{norm} \simeq 0.8957$$

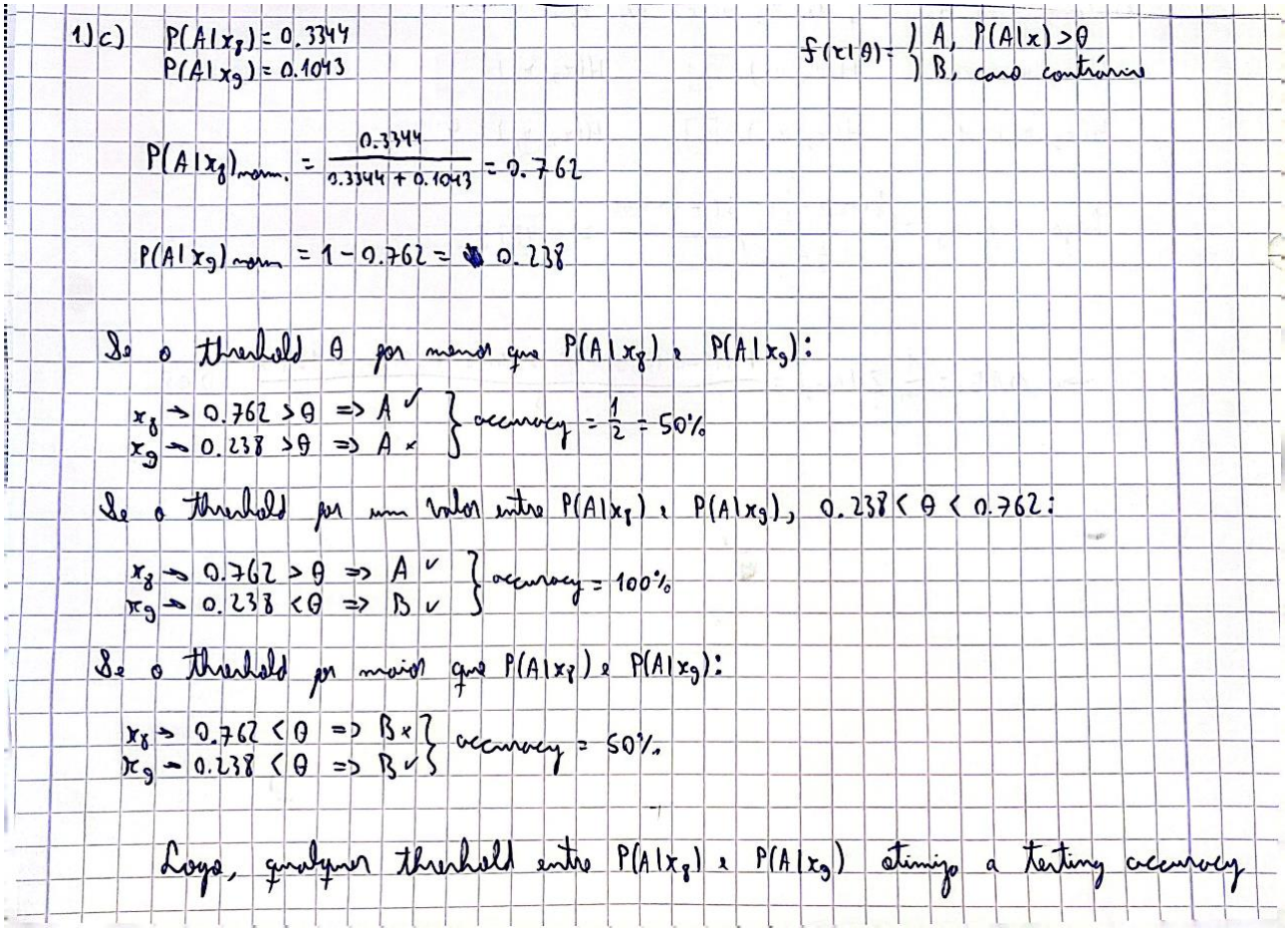
Logo, também classificamos x9 como B

Aprendizagem 2022/23

Homework II – Group 107

(ist1103811, ist1103479)

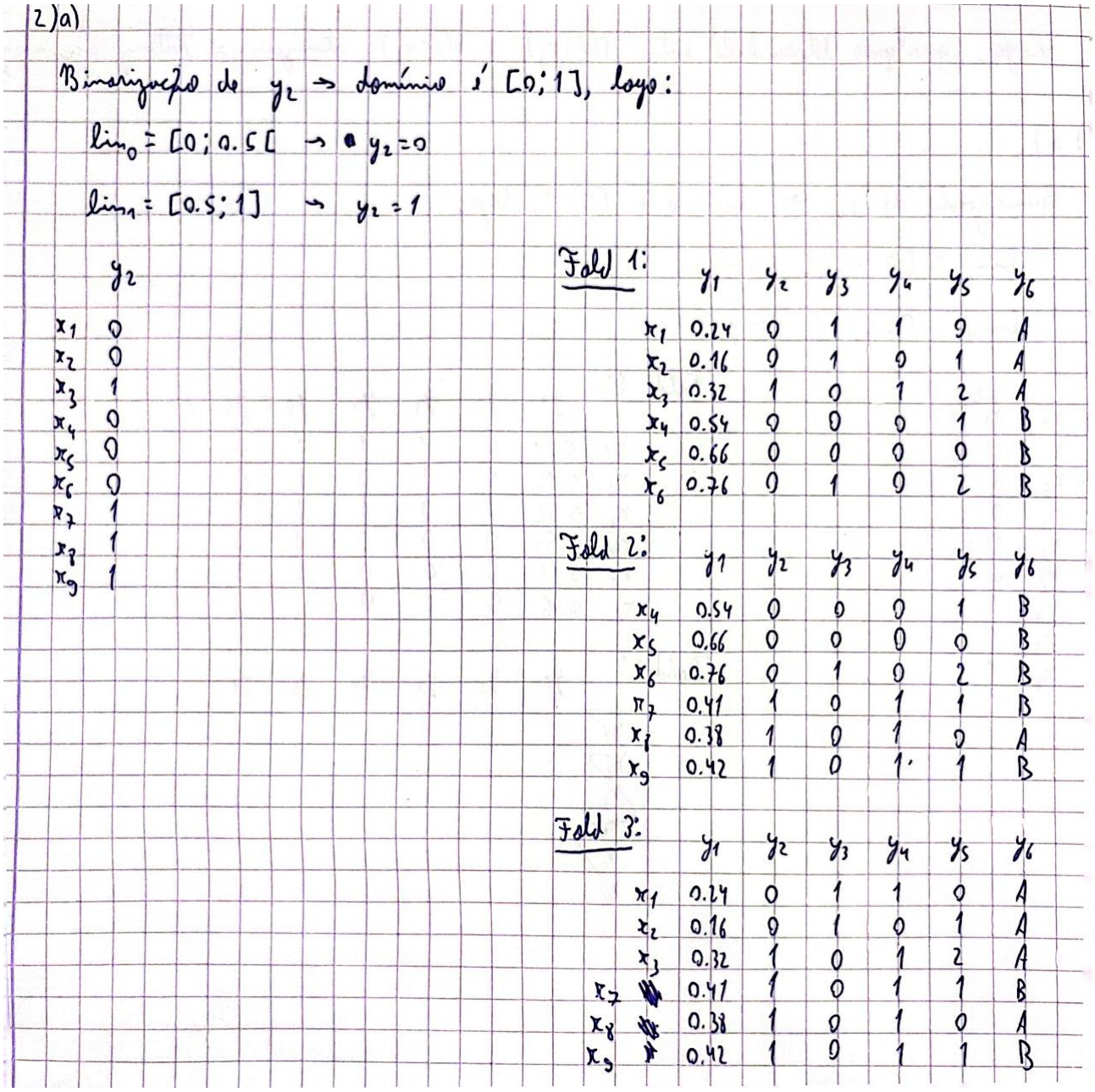
1)c)



Homework II – Group 107

(ist1103811, ist1103479)

2)a)



2)a)

Binarização de y_2 → domínio é [0; 1], logo:

$lim_0 = [0; 0.5[\rightarrow y_2 = 0$

$lim_1 = [0.5; 1] \rightarrow y_2 = 1$

	y_2
x_1	0
x_2	0
x_3	1
x_4	0
x_5	0
x_6	0
x_7	1
x_8	1
x_9	1

Fold 1:

	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

Fold 2:

	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
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

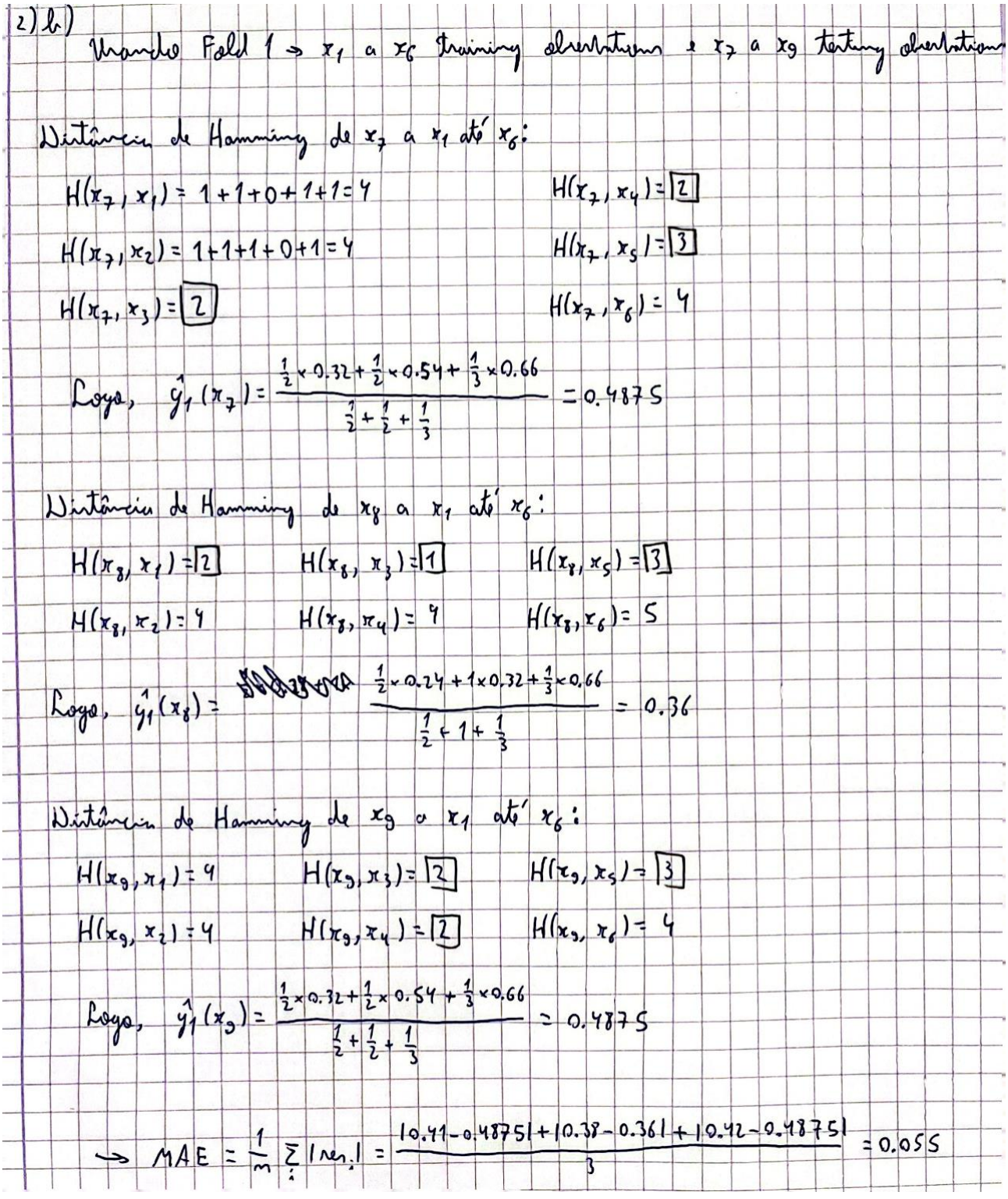
Fold 3:

	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_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

Homework II – Group 107

(ist1103811, ist1103479)

2)b)



Homework II – Group 107

(ist1103811, ist1103479)

II. Programming and critical analysis

1)a)

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import arff
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from scipy.stats import ttest_rel
from sklearn.preprocessing import LabelEncoder

# Load the ARFF dataset
data, meta = arff.loadarff('column_diagnosis.arff')

# Convert the dataset to a Pandas DataFrame
import pandas as pd
df = pd.DataFrame(data)

# Extract features (X) and encode the target labels (y)
X = df.drop(columns=['class']).values
y = df['class'].str.decode('utf-8')  # Decoding binary strings
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(y)

# Set random seed for reproducibility
random_seed = 0

# Define classifiers
knn_classifier = KNeighborsClassifier(n_neighbors=5)
nb_classifier = GaussianNB()

# Define stratified 10-fold cross-validation
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=random_seed)

# Perform cross-validation and collect accuracy scores
knn_scores = cross_val_score(knn_classifier, X, y, cv=cv, scoring='accuracy')
nb_scores = cross_val_score(nb_classifier, X, y, cv=cv, scoring='accuracy')

# Plot boxplots of the fold accuracies
plt.figure(figsize=(10, 6))
plt.boxplot([knn_scores, nb_scores], labels=['kNN', 'Naïve Bayes'])
plt.title('Accuracy Comparison: kNN vs. Naïve Bayes')
plt.ylabel('Accuracy')
plt.savefig("Ex1ab")
```

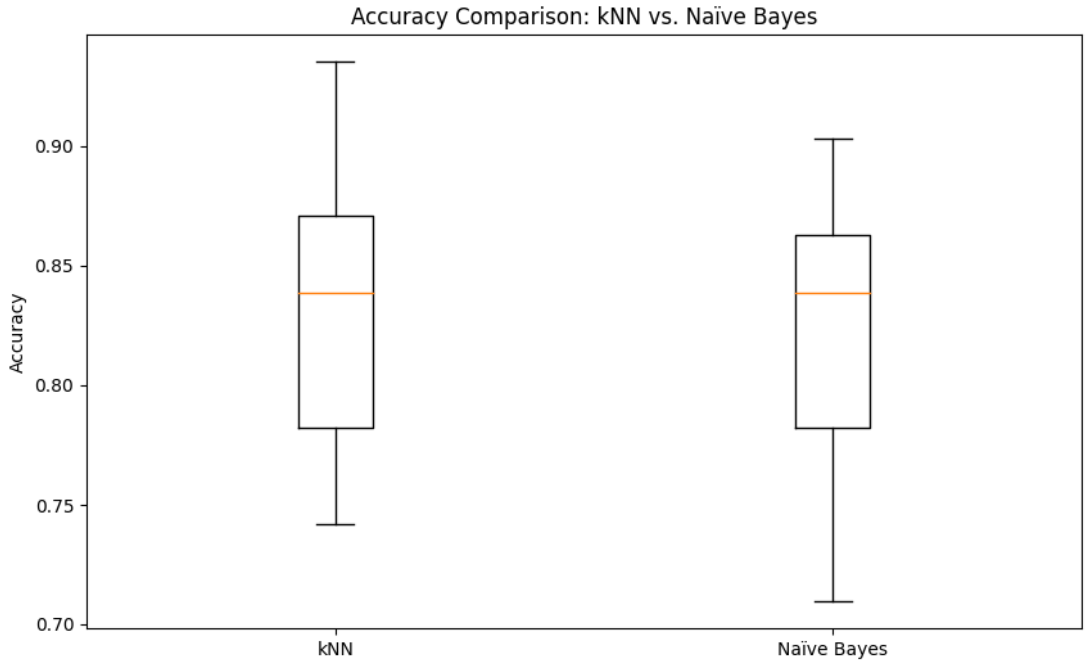
Homework II – Group 107

(ist1103811, ist1103479)

```
# Perform a statistical test to compare the classifiers
t_statistic, p_value = ttest_rel(knn_scores, nb_scores)

# Set the significance level
alpha = 0.05

if p_value < alpha:
    print(f"kNN is statistically superior to Naïve Bayes (p-
value={p_value:.4f})")
else:
    print(f"There is no statistically significant difference between kNN and
Naïve Bayes (p-value={p_value:.4f})")
```



1)b)

There is no statistically significant difference between kNN and Naïve Bayes (p-value=0.3809)

2)

```
import pandas as pd
from scipy.io import arff
from sklearn.model_selection import cross_val_predict, StratifiedKFold
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the ARFF dataset
data = arff.loadarff('column_diagnosis.arff')
df = pd.DataFrame(data[0])

# Ensure the 'class' column contains only valid class labels
df['class'] = df['class'].str.decode('utf-8')  # Convert bytes to strings

# Encode the 'class' column to numerical values
le = LabelEncoder()
df['class'] = le.fit_transform(df['class'])

# Split the dataset into features (X) and the target variable (y)
X = df.drop('class', axis=1)  # Features
y = df['class']  # Target variable

# Create k-NN classifiers with k = 1 and k = 5
knn_1 = KNeighborsClassifier(n_neighbors=1)
knn_5 = KNeighborsClassifier(n_neighbors=5)

# Initialize StratifiedKFold with 10 folds and shuffling
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=0)

# Perform cross-validation and get predicted labels for each fold for k=1 and k=5
predicted_labels_1 = cross_val_predict(knn_1, X, y, cv=cv)
predicted_labels_5 = cross_val_predict(knn_5, X, y, cv=cv)

# Get the unique class names
class_names = le.classes_

# Calculate confusion matrices for both k-NN classifiers
confusion_matrix_1 = pd.crosstab(index=y, columns=predicted_labels_1,
rownames=['True'], colnames=['Predicted'])
confusion_matrix_5 = pd.crosstab(index=y, columns=predicted_labels_5,
rownames=['True'], colnames=['Predicted'])
```

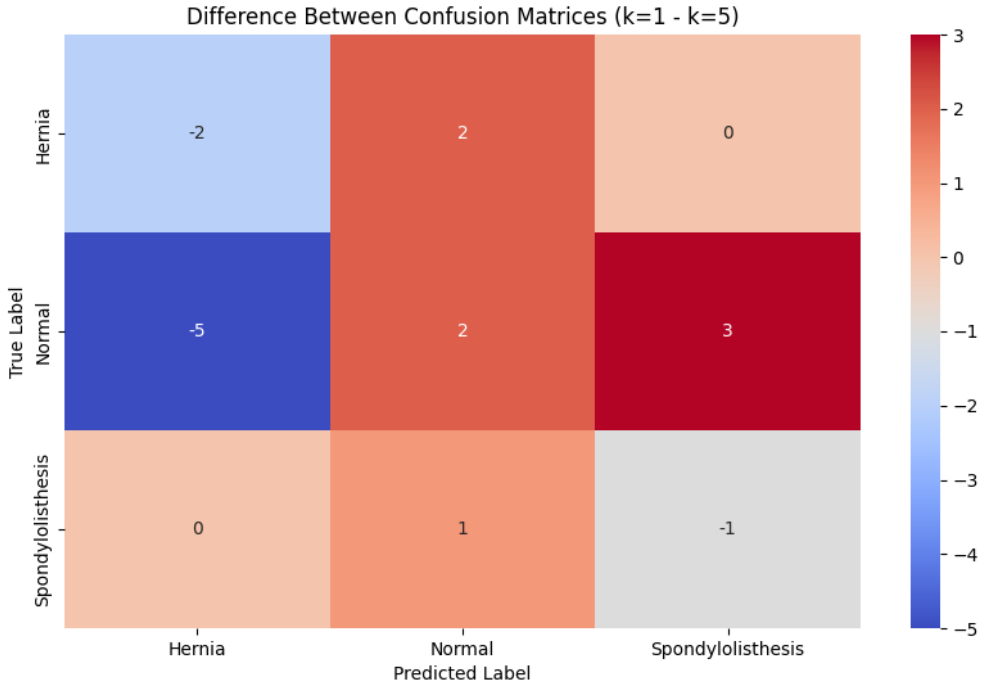
Homework II – Group 107

(ist1103811, ist1103479)

```python
# Calculate the difference between the two confusion matrices
difference_matrix = confusion_matrix_1 - confusion_matrix_5

# Print the difference matrix
print("Matrix k=1:\n")
print(confusion_matrix_1)
print("Matrix k=5:\n")
print(confusion_matrix_5)
print("\nDifference Matrix (k=1 - k=5):")
print(difference_matrix)

# Create a heatmap to visualize the differences
plt.figure(figsize=(10, 6))
sns.heatmap(difference_matrix, annot=True, fmt="d", cmap="coolwarm", cbar=True)
plt.title('Difference Between Confusion Matrices (k=1 - k=5)')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.xticks(np.arange(len(class_names)) + 0.5, class_names)
plt.yticks(np.arange(len(class_names)) + 0.5, class_names)
plt.savefig("Ex2")
```



Analisando os resultados vemos que quanto mais positivos forem os valores das células , concluimos que um k=1 é melhor que um k=5, no que toca a estimar a class do atributo em questão. Por outro

lado, se o valor da célula for negativo tiramos uma conclusão semelhante, onde aqui o k preferível é o k=5. Nas células cinzentas , neutras sabemos que tanto o k=1 e o k=5 têm a mesma performance, sendo igual escolher um ou outro.

Tendo em conta esta observação concluimos que não existe um k que seja universalmente ótimo, sendo preciso ter sempre em consideração vários dados, como as características dos dados, a natureza das classes, etc.

3)

Tendo em conta column_diagnosis, podemos identificar 3 possíveis dificuldades:

1. Suposição de Independência de Características: O Naïve Bayes assume que as características são condicionalmente independentes dadas as classes. No entanto, no conjunto de dados "column_diagnosis", algumas características podem não ser inteiramente independentes. Por exemplo, medidas relacionadas à saúde da coluna, como 'pelvic_incidence' e 'pelvic_tilt', podem estar correlacionadas. Essa suposição de independência pode não ser verdadeira, o que pode afetar o desempenho do modelo.

2. Características Contínuas: O Naïve Bayes foi projetado para lidar com dados discretos e características categóricas. Se o conjunto de dados contém características contínuas, como é comum em conjuntos de dados biomédicos, pode ser necessário discretizá-las para encaixá-las no framework do Naïve Bayes. A escolha do método de discretização pode afetar o desempenho do modelo, e uma discretização inadequada pode resultar na perda de informações.

3. Limitação na Expressividade do Modelo: O Naïve Bayes é um classificador simples e linear. Ele pode não capturar relações complexas entre características nos dados. No conjunto de dados "column_diagnosis", pode haver padrões não lineares ou intrincados que o Naïve Bayes tem dificuldade em modelar eficazmente. Modelos mais avançados, como árvores de decisão, florestas aleatórias ou máquinas de vetores de suporte, podem ser mais adequados para capturar essas relações.

END