Aprendizagem 2023
Homework I — Group 003
(ist1107028, ist1107137)

Part I: Pen and paper

We collected four positive (P) observations,

$$\{\boldsymbol{x_1 = (A, 0)}, \quad \boldsymbol{x_2 = (B, 1)}, \quad \boldsymbol{x_3 = (A, 1)}, \quad \boldsymbol{x_4 = (A, 0)}\}$$

and four negative (N) observations,

$$\{\boldsymbol{x_5 = (B, 0)}, \quad \boldsymbol{x_6 = (B, 0)}, \quad \boldsymbol{x_7 = (A, 1)}, \quad \boldsymbol{x_8 = (B, 1)}\}$$

Consider the problem of classifying observations as positive or negative.

1. **Compute the F1-measure of a kNN with k = 5 and Hamming distance using a leave-one-out evaluation schema. Show all calculus.**

We start by calculating Hamming distance between observations. The Hamming distance is the number of positions at which the corresponding symbols are different.
Since we are workin with $k = 5$, we will consider the 5 nearest neighbors of each observation (written in blue).

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	-	2	**1**	**0**	**1**	**1**	**1**	2
x_2	2	-	**1**	2	**1**	**1**	**1**	**0**
x_3	**1**	**1**	-	**1**	2	2	**0**	**1**
x_4	**0**	2	**1**	-	**1**	**1**	**1**	2
x_5	**1**	**1**	2	**1**	-	**0**	2	**1**
x_6	**1**	**1**	2	**1**	**0**	-	2	**1**
x_7	**1**	**1**	**0**	**1**	2	2	-	**1**
x_8	2	**0**	**1**	2	**1**	**1**	**1**	-

Table 1: Hamming distance between observations

Now that we have the Hamming distance between all observations, we must identify if the prediction is correct or not. We will consider the majority class of the 5 nearest neighbors for each observation.

Example: For x_1, the 5 nearest neighbors are x_3 and x_4 (which are positive), x_5, x_6 and x_7 (which are negative). The majority class is negative, therefore the prediction is incorrect.

We apply the same logic for the rest of the classes, ending up with the following table:

Observation	True Value	Prediction	Confusion Matrix Terminology
x_1	P	N	FN
x_2	P	N	FN
x_3	P	P	TP
x_4	P	N	FN
x_5	N	P	FP
x_6	N	P	FP
x_7	N	P	FP
x_8	N	N	TN

P - Positive observation; N - Negative observation

TP - True Positive; TN - True Negative; FP - False Positive; FN - False Negative

Table 2: Predictions for each observation - $k = 5$

With this table, we can know calculate the Precision, Recall and F1-measure using the following formulas:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{1}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2}$$

$$\text{F1-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

Replacing the corresponding values in the formulas, we get:

for Precision (1) and Recall (2):

$$\text{Precision} = \frac{1}{1+3} = 0.25 \qquad \text{Recall} = \frac{1}{1+3} = 0.25$$

F1-measure (3):

$$\text{F1-measure} = 2 \times \frac{0.25 \times 0.25}{0.25 + 0.25} = 0.25$$

2. **Propose a new metric (distance) that improves the latter's performance (i.e., the F1-measure) by three fold.**

   To improve the F1-measure, we propose using $K = 3$ instead of $k = 5$. This way, we will consider the 3 nearest neighbors of each observation.
   Simply by looking at the Hamming distance table, we can produce a new prediton table, like in the first exercise:

Observation	True Value	Prediction	Confusion Matrix Terminology
x_1	P	P	TP
x_2	P	N	FN
x_3	P	N	FN
x_4	P	P	TP
x_5	N	N	TN
x_6	N	N	TN
x_7	N	P	FP
x_8	N	P	FP

P - Positive observation; N - Negative observation

TP - True Positive; TN - True Negative; FP - False Positive; FN - False Negative

Table 3: Predictions for each observation - $k = 3$

$$\text{Precision} = \frac{2}{2+2} = 0.5 \qquad \text{Recall} = \frac{2}{2+2} = 0.5$$

$$\text{F1-measure} = 2 \times \frac{0.5 \times 0.5}{0.5 + 0.5} = 0.5$$

An additional positive observation was acquired, $x_9 = (B, 0)$, and a third variable y_3 was independently monitored, yielding estimates,

$$\boldsymbol{y_3|P = \{1.1, 0.8, 0.5, 0.9, 0.8\} \quad and \quad y_3|N = \{1, 0.9, 1.2, 0.9\}}$$

3. **Considering the nine training observations, learn a Bayesian classifier assuming: (i) y_1 and y_2 are dependent; (ii) y_1, y_2 and y_3 variable sets are independent and equally important; and (iii) y_3 is normally distributed. Show all parameters.**

   With the nine training observations, we can calculate the parameters for the Bayesian classifier. We will refer to the outcome, which can be positive or negative, as P and N respectively, and 'class' or 'c' when we mean both.

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

$$P(\text{class}|y_1, y_2, y_3) = \frac{P(y_1, y_2, y_3|\text{class}) \times P(\text{class})}{P(y_1, y_2, y_3)} \tag{4}$$

Since we know $\{y_1, y_2\}$ and $\{y_3\}$ are independent, we can rewrite $P(y_1, y_2, y_3)$ as $P(y_1, y_2) \times P(y_3)$. Rewriting (4) with this, results in:

$$P(\text{class}|y_1, y_2, y_3) = \frac{P(y_1, y_2|\text{class}) P(y_3|\text{class}) \times P(\text{class})}{P(y_1, y_2) \times P(y_3)} \tag{5}$$

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.

$$\begin{aligned}\hat{z} &= \underset{c\in\{P,N\}}{\arg\max}\ \{P(\text{c}|O)\} \\ &= \underset{c\in\{P,N\}}{\arg\max} \left\{\frac{P(y_1, y_2|c)P(y_3|c) \times P(c)}{P(y_1, y_2)P(y_3)}\right\} \\ &= \underset{c\in\{P,N\}}{\arg\max}\ \{P(y_1, y_2|c)P(y_3|c) \times P(c)\} \quad \text{(we can remove parameters that do not depend on } c\text{)}\end{aligned} \tag{6}$$

We can now begin to compute these parameters.

$$\text{Priors:} \qquad P(P) = \frac{5}{9} \qquad P(N) = 1 - P(P) = \frac{4}{9}$$

To get the likelihoods, we start by calculating the joint probabilities for the categorical variables y_1 and y_2:

$$\begin{aligned} P((y_1 = A, y_2 = 0)|P) &= \frac{2}{5} & P((y_1 = A, y_2 = 1)|P) &= \frac{1}{5} \\ P((y_1 = A, y_2 = 0)|N) &= \frac{0}{4} & P((y_1 = A, y_2 = 1)|N) &= \frac{1}{4} \\ P((y_1 = B, y_2 = 0)|P) &= \frac{1}{5} & P((y_1 = B, y_2 = 1)|P) &= \frac{1}{5} \\ P((y_1 = B, y_2 = 0)|N) &= \frac{2}{4} & P((y_1 = B, y_2 = 1)|N) &= \frac{1}{4} \end{aligned}$$

For the third variable, we know that it is normally distributed, so we start by calculating the mean and variance for each class:

$$\textbf{mean:} \quad \mu = \frac{1}{n}\sum_{i=1}^{n} y_i \qquad\qquad \textbf{variance:} \quad \sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \mu)^2$$

Positive class:

$$\begin{aligned}
\mu_P &= \tfrac{1.1+0.8+0.5+0.9+0.8}{5} = 0.82 \\
\sigma_P^2 &= \tfrac{(1.1-0.82)^2+(0.8-0.82)^2+(0.5-0.82)^2+(0.9-0.82)^2+(0.8-0.82)^2}{4} = 0.47
\end{aligned}$$

Negative class:

$$\begin{aligned}
\mu_N &= \tfrac{1+0.9+1.2+0.9}{4} = 1.0 \\
\sigma_N^2 &= \tfrac{(1-1)^2+(0.9-1)^2+(1.2-1)^2+(0.9-1)^2}{3} = 0.02
\end{aligned}$$

Since y_3 is normally distributed, we can calculate the likelihood for each class using the normal distribution formula:

$$P(y_z|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_z-\mu)^2}{2\sigma^2}\right) \tag{7}$$

We must now replace y_3 with each value in the set $\{1.1, 0.8, 0.5, 0.9, 0.8\}$ and $\{1, 0.9, 1.2, 0.9\}$ to calculate the likelihoods for each class. Having the parameters for the normal distribution, we can calculate the likelihoods for y_3 as per (7):

Positive class:

$$\begin{aligned}
P(y_3 = 1.1|\mu_P, \sigma_P^2) &= \frac{1}{\sqrt{2\pi \times 0.47}} \exp\left(-\frac{(1.1-0.82)^2}{2 \times 0.47}\right) \approx 0.535 \\
P(y_3 = 0.8|\mu_P, \sigma_P^2) &= \frac{1}{\sqrt{2\pi \times 0.47}} \exp\left(-\frac{(0.8-0.82)^2}{2 \times 0.47}\right) \approx 0.582 \\
P(y_3 = 0.5|\mu_P, \sigma_P^2) &= \frac{1}{\sqrt{2\pi \times 0.47}} \exp\left(-\frac{(0.5-0.82)^2}{2 \times 0.47}\right) \approx 0.522 \\
P(y_3 = 0.9|\mu_P, \sigma_P^2) &= \frac{1}{\sqrt{2\pi \times 0.47}} \exp\left(-\frac{(0.9-0.82)^2}{2 \times 0.47}\right) \approx 0.578
\end{aligned}$$

Negative class:

$$\begin{aligned}
P(y_3 = 1|\mu_N, \sigma_N^2) &= \frac{1}{\sqrt{2\pi \times 0.02}} \exp\left(-\frac{(1-1)^2}{2 \times 0.02}\right) \approx 2.821 \\
P(y_3 = 0.9|\mu_N, \sigma_N^2) &= \frac{1}{\sqrt{2\pi \times 0.02}} \exp\left(-\frac{(0.9-1)^2}{2 \times 0.02}\right) \approx 2.197 \\
P(y_3 = 1.2|\mu_N, \sigma_N^2) &= \frac{1}{\sqrt{2\pi \times 0.02}} \exp\left(-\frac{(1.2-1)^2}{2 \times 0.02}\right) \approx 1.038
\end{aligned}$$

Now we have all the parameters to apply the Bayesian classifier to new observations.

Consider now three testing observations,

$$\{(\boldsymbol{A}, \mathbf{1}, \mathbf{0.8}), (\boldsymbol{B}, \mathbf{1}, \mathbf{1}), (\boldsymbol{B}, \mathbf{0}, \mathbf{0.9})\}$$

4. **Under a MAP assumption, classify each testing observation showing all your calculus.**

MAP (Maximum A Posteriori) is defined as:

$$\begin{aligned}\hat{z} &= \arg\max_{c_i} \{P(c_i|x)\} \\ &= \arg\max_{c_i} \left\{\frac{P(x|c_i) \times P(c_i)}{P(x)}\right\} \\ &= \arg\max_{c_i} \{P(x|c_i) \times P(c_i)\}\end{aligned} \tag{8}$$

In order to apply this assumption we use the likelihoods and the priors calculated in the beginning of this report.

$$P(P) = \frac{5}{9} \qquad P(N) = 1 - P(P) = \frac{4}{9}$$

$$\begin{aligned}P((y_1 = A, y_2 = 1)|P) &= \frac{1}{5} \quad P((y_1 = A, y_2 = 1)|N) = \frac{1}{4} \\ P((y_1 = B, y_2 = 1)|P) &= \frac{1}{5} \quad P((y_1 = B, y_2 = 1)|N) = \frac{1}{4} \\ P((y_1 = B, y_2 = 0)|P) &= \frac{1}{5} \quad P((y_1 = B, y_2 = 0)|N) = \frac{2}{4}\end{aligned}$$

$$\begin{aligned}P(y_3 = 0.8|\mu_P, \sigma_P^2) &\approx 0.582 \quad P(y_3 = 1.0|\mu_N, \sigma_N^2) \approx 2.821 \\ P(y_3 = 0.9|\mu_N, \sigma_N^2) &\approx 2.197 \quad P(y_3 = 0.9|\mu_P, \sigma_P^2) \approx 0.578\end{aligned}$$

Before using MAP assumption, we must calculate the missing values needed - $P(y_3 = 0.8|\mu_N, \sigma_N^2)$ and $P(y_3 = 1|\mu_P, \sigma_P^2)$.

$$\begin{aligned}P(y_3 = 0.8|\mu_N, \sigma_N^2) &= \frac{1}{\sqrt{2\pi \times 0.02}} \exp\left(-\frac{(0.8-1)^2}{2 \times 0.02}\right) \approx 2.821 \\ P(y_3 = 1|\mu_P, \sigma_P^2) &= \frac{1}{\sqrt{2\pi \times 0.47}} \exp\left(-\frac{(1-0.82)^2}{2 \times 0.47}\right) \approx 0.535\end{aligned}$$

With this, we can just replace the values in the MAP formula (8) to get the predictions for each observation.

$\boxed{(A, 1, 0.8)}$

$$\begin{aligned} P(y_1 = A, y_2 = 1, y_3 = 0.8|P) &= P(y_1 = A, y_2 = 1|\text{P}) \cdot P(y_3 = 0.8|\text{P}) \cdot P(P) \\ &= \frac{1}{5} \times 0.582 \times \frac{5}{9} \approx 0.065 \end{aligned}$$

$$\begin{aligned} P(y_1 = A, y_2 = 1, y_3 = 0.8|N) &= P(y_1 = A, y_2 = 1|\text{N}) \times P(y_3 = 0.8|\text{N}) \times P(N) \\ &= \frac{1}{4} \times 2.197 \times \frac{4}{9} \approx 0.488 \end{aligned}$$

$$\begin{aligned} \hat{z}_{(A,1,0.8)} &= \underset{c \in \{P,N\}}{\arg\max} \; \{P(y_1 = A, y_2 = 1|c)P(y_3 = 0.8|c) \times P(c)\} \\ &= \arg\max \; \{P(y_1, y_2|P)P(y_3|P) \times P(P); P(y_1, y_2|N)P(y_3|N) \times P(N)\} \\ &= N \end{aligned}$$

$\boxed{(B, 1, 1)}$

$$\begin{aligned} P(y_1 = B, y_2 = 1, y_3 = 1|P) &= P(y_1 = B, y_2 = 1|\text{P}) \times P(y_3 = 1|\text{P}) \times P(P) \\ &= \frac{1}{5} \times 2.821 \times \frac{5}{9} \approx 0.313 \end{aligned}$$

$$\begin{aligned} P(y_1 = B, y_2 = 1, y_3 = 1|N) &= P(y_1 = B, y_2 = 1|\text{N}) \times P(y_3 = 1|\text{N}) \times P(N) \\ &= \frac{1}{4} \times 0.535 \times \frac{4}{9} \approx 0.059 \end{aligned}$$

$$\begin{aligned} \hat{z}_{(B,1,1)} &= \underset{c \in \{P,N\}}{\arg\max} \; \{P(y_1 = B, y_2 = 1|c)P(y_3 = 1|c) \times P(c)\} \\ &= \arg\max \; \{P(y_1, y_2|P)P(y_3|P) \times P(P); P(y_1, y_2|N)P(y_3|N) \times P(N)\} \\ &= P \end{aligned}$$

$\boxed{(B, 0, 0.9)}$

$$\begin{aligned} P(y_1 = B, y_2 = 0, y_3 = 0.9|P) &= P(y_1 = B, y_2 = 0|\text{P}) \times P(y_3 = 0.9|\text{P}) \times P(P) \\ &= \frac{1}{5} \times 0.578 \times \frac{5}{9} \approx 0.064 \end{aligned}$$

$$\begin{aligned} P(y_1 = B, y_2 = 0, y_3 = 0.9|N) &= P(y_1 = B, y_2 = 0|\text{N}) \times P(y_3 = 0.9|\text{N}) \times P(N) \\ &= \frac{2}{4} \times 2.197 \times \frac{4}{9} \approx 0.488 \end{aligned}$$

$$\begin{aligned} \hat{z}_{(B,0,0.9)} &= \underset{c \in \{P,N\}}{\arg\max} \; \{P(y_1 = B, y_2 = 0|c)P(y_3 = 0.9|c) \times P(c)\} \\ &= \arg\max \; \{P(y_1, y_2|P)P(y_3|P) \times P(P); P(y_1, y_2|N)P(y_3|N) \times P(N)\} \\ &= N \end{aligned}$$

Therefore, the predictions for each observation are N, P and N respectively.

At last, consider only the following sentences and their respective connotations,

$$\{(\text{"}\textit{Amazing run}\text{"}, P), (\text{"}\textit{I like it}\text{"}, P), (\text{"}\textit{To tired}\text{"}, N), (\text{"}\textit{Bad run}\text{"}, N)\}$$

5. Using a naïve Bayes under a ML assumption, classify the new sentence "I like to run". For the likelihoods calculation consider the following formula,

$$P(T_i|c) = \frac{freq(t_i) + 1}{N_c + V}$$

where t_i represents a certain term i, V the number of unique terms in the vocabulary, and N_c the total number of terms in class c. Show all calculus.

Part II: Programming

Consider the heart-disease.csv dataset available at the course webpage's homework tab. Using sklearn, apply a 5-fold stratified cross-validation with shuffling (random_state = 0) for the assessment of predictive models along this section.

(1) Compare the performance of a kNN with k = 5 and a Naive Bayes with Gaussian assumption (consider all remaining parameters as default):

a. Plot two boxplots with the fold accuracies for each classifier. Is there one more stable than the other regarding performance? Why do you think that is the case? Explain.

```python
import matplotlib.pyplot as plt, pandas as pd
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB

# Read the dataset
df = pd.read_csv("./heart-disease.csv")
X, y = df.drop("target", axis=1), df["target"]

# Define cross-validation strategy
folds = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)

# Initialize classifiers
knn_predictor = KNeighborsClassifier(n_neighbors=5)
nb_predictor = GaussianNB()

# Evaluate classifiers
knn_accs = cross_val_score(knn_predictor, X, y, cv=folds, scoring="accuracy")
nb_accs = cross_val_score(nb_predictor, X, y, cv=folds, scoring="accuracy")

# Plot boxplots
plt.figure(figsize=(7, 5))
b_plot = plt.boxplot(
    [knn_accs, nb_accs], patch_artist=True, labels=["kNN", "Naive Bayes"]
)

colors = ["#1f77b4", "#E40071"]
for patch, color in zip(b_plot["boxes"], colors):
    patch.set_facecolor(color)
for median in b_plot["medians"]:
    median.set_color("black")

plt.ylabel("Accuracy")
plt.grid(axis="y")
plt.show()
```

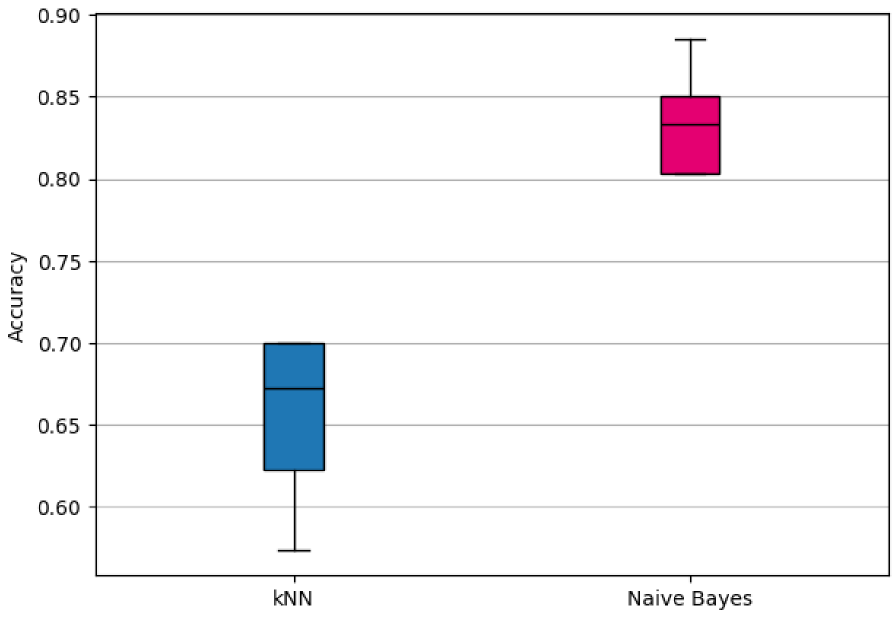

Figure 1: Boxplot of the fold accuracies for each classifier

b. Report the accuracy of both models, this time scaling the data with a Min-Max scaler before training the models. Explain the impact that this preprocessing step has on the performance of each model, providing an explanation for the results.

```python
import matplotlib.pyplot as plt, pandas as pd
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline

# Read the ARFF file and prepare data
df = pd.read_csv("./heart-disease.csv")
X, y = df.drop("target", axis=1), df["target"]

# Define cross-validation strategy
folds = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)
```

```python

# Create pipelines for scaling
knn_pipeline = Pipeline([("scaler", MinMaxScaler()), ("knn",
    KNeighborsClassifier(n_neighbors=5))])
nb_pipeline = Pipeline([("scaler", MinMaxScaler()), ("nb", GaussianNB())])

# Evaluate classifiers
knn_accs = cross_val_score(knn_pipeline, X, y, cv=folds, scoring="accuracy")
nb_accs = cross_val_score(nb_pipeline, X, y, cv=folds, scoring="accuracy")

# Plot boxplots
plt.figure(figsize=(7, 5))
b_plot = plt.boxplot(
    [knn_accs, nb_accs], patch_artist=True, labels=["kNN", "Naive Bayes"]
)

colors = ["#1f77b4", "#E40071"]
for patch, color in zip(b_plot["boxes"], colors):
    patch.set_facecolor(color)
for median in b_plot["medians"]:
    median.set_color("black")

plt.ylabel("Accuracy")
plt.grid(axis="y")
plt.show()
```

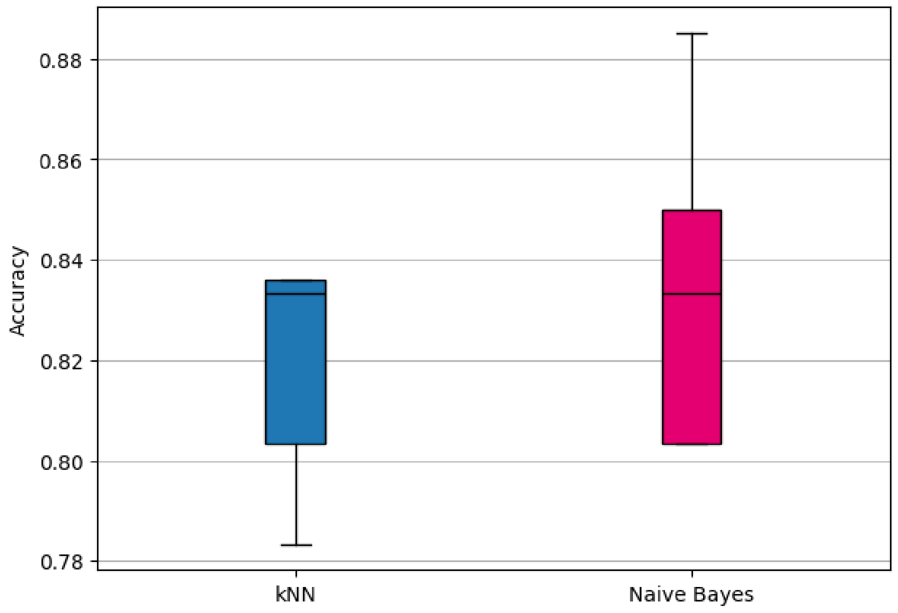

Figure 2: Accuracy of both models with Min-Max scaling

c. Using scipy, test the hypothesis ”the kNN model is statistically superior to naive Bayes regarding accuracy”, asserting whether it is true.

(2) a 80-20 train-test split, vary the number of neighbors of a kNN classifier using

k = $\{1, 5, 10, 20, 30\}$. Additionally, for each k, train one classifier using uniform weights and distance weights.

a. Plot the train and test accuracy for each model

```python
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Read the dataset
df = pd.read_csv("./heart-disease.csv")
X, y = df.drop("target", axis=1), df["target"]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=0)

# Standardize the features
scaler = StandardScaler().fit(X_train)
X_train_scaled, X_test_scaled = scaler.transform(X_train), scaler.transform(
    X_test)

# Define different values of k
number_of_neighbors = [1, 5, 10, 20, 30]

# Initialize lists to store accuracy results
train_acc_uniform, test_acc_uniform = [], []
train_acc_distance, test_acc_distance = [], []

# Train and evaluate KNN with 'uniform' weights
for k in number_of_neighbors:
    knn_uniform = KNeighborsClassifier(n_neighbors=k, weights="uniform")
    knn_uniform.fit(X_train_scaled, y_train)
    train_acc_uniform.append(knn_uniform.score(X_train_scaled, y_train))
    test_acc_uniform.append(knn_uniform.score(X_test_scaled, y_test))

# Train and evaluate KNN with 'distance' weights
for k in number_of_neighbors:
    knn_distance = KNeighborsClassifier(n_neighbors=k, weights="distance")
    knn_distance.fit(X_train_scaled, y_train)
    train_acc_distance.append(knn_distance.score(X_train_scaled, y_train))
    test_acc_distance.append(knn_distance.score(X_test_scaled, y_test))

# Plot the results
plt.figure(figsize=(16, 6))

# Plot distance accuracy
plt.subplot(1, 2, 1)
plt.plot(
    number_of_neighbors,
    test_acc_distance,
    label='Test Accuracy',
    marker='D',
    color='#E40071'
)

plt.plot(
    number_of_neighbors,
    train_acc_distance,
    label='Train Accuracy',
    marker='o',
    color='#1f77b4')

plt.xlabel('Number of neighbors')
plt.ylabel('Distance')
plt.title('KNN Accuracy with Distance Weights')
plt.grid(True)
plt.legend()

# Plot uniform accuracy
plt.subplot(1, 2, 2)
plt.plot(
    number_of_neighbors,
    test_acc_uniform,
    label='Test Accuracy',
    marker='D',
    color='#E40071'
)

plt.plot(
    number_of_neighbors,
    train_acc_uniform,
    label='Train Accuracy',
    marker='o',
    color='#1f77b4'
)

# Add labels and title
plt.xlabel('Number of neighbors')
plt.ylabel('Uniform')
plt.title('KNN Accuracy with Uniform Weights')
plt.grid(True)
plt.legend()
plt.show()
```

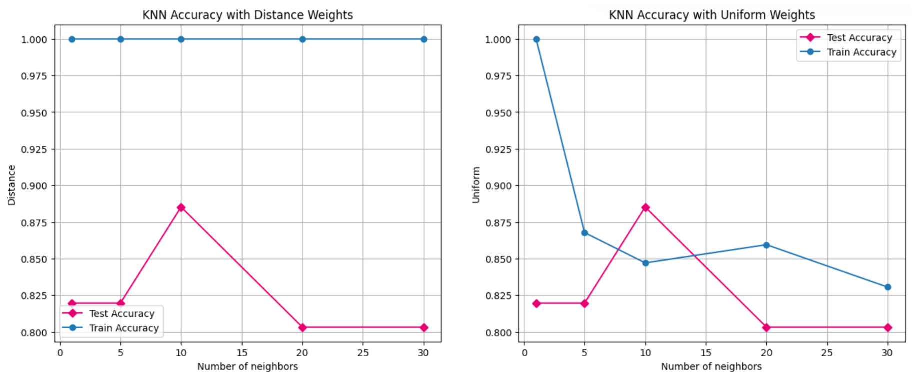

Figure 3: Train and test accuracy for each model

b. Explain the impact of increasing the neighbors on the generalization ability of the models.

(3) Considering the unique properties of the heart-disease.csv dataset, identify two possible difficulties of the naïve Bayes model used in the previous exercises when learning from the given dataset.