Aprendizagem 2022/23

Homework II – Group 51

(ist1105940, ist1106221)

I. Pen-and-paper

1) [3.0v] Compute the F1-measure of a NN with $k = 5$ and Hamming distance using a leave-one-out evaluation schema. Show all calculus.

1) Distancias para x_i distância de Hamming 1 se 0 se 2 parametros iguais 1 se um parametro igual 2 se 0 parametros iguais

	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			1	2	1	1	1	0
x_3				1	2	2	0	1
x_4					1	1	1	2
x_5						0	2	1
x_6							2	1
x_7								1
x_8								

vizinhos de x_1: $x_4; x_3; x_5; x_6; x_7$
x_2: $x_8; x_3; x_5; x_6; x_7$
x_3: $x_7; x_1; x_2; x_4; x_8$
x_4: $x_1; x_3; x_5; x_6; x_7$
x_5: $x_1; x_2; x_4; x_6; x_8$
x_6: $x_1; x_2; x_4; x_5; x_8$
x_7: $x_1; x_2; x_3; x_4; x_8$
x_8: $x_2; x_3; x_5; x_6; x_7$

% classificação para cada x_i de acordo com os seus vizinhos

x_1 2P 3N → N
x_2 1P 4N → N
x_3 3P 2N → P
x_4 2P 3N → N
x_5 3P 2N → P
x_6 3P 2N → P
x_7 4P 1N → P
x_8 2P 3N → N

Homework II – Group 51

(ist1105940, ist1106221)

matriz de confusão

Predicted \ Real	P	N
P	1	3
N	3	1

$$P = \frac{TP}{TP+FP} \qquad R = \frac{TP}{TP+FN} \qquad F_{1\,score} = \frac{2}{\frac{1}{P}+\frac{1}{R}}$$

$$P = \frac{1}{1+3} = 0.25 \qquad R = \frac{1}{1+3} = 0.25$$

$$F_1 = \frac{2}{\frac{1}{0.25}+\frac{1}{0.25}} = \frac{2}{\frac{2}{0.25}} = 0.25$$

Homework II – Group 51

(ist1105940, ist1106221)

2) [2.5v] Propose a new metric (distance and/or) that improves the latter's performance (i.e.,*k* the F1-measure) by three fold.

2) para que o F1 score aumente 3 x a nossa matriz de confusão será do tipo

Real

Predicted	P	N
P	3	1
N	1	3

ficando o F1score = $\frac{3}{4}$

decidi usar uma distancia pesada do tipo

$d(x_i x_j) = w_1 (*y_{1i} - y_{1j}) + w_2 (y_{2i} - y_{2j})$

usando distancias de hamming para y_1 e y_2

e os pesos $w_1 = 1.10$ e $w_2 = 0.10$

e $k = 3$

Usamos $k=3$ porque existem 4 positivos e 4 negativos e quando estamos a verificar se uma instância é de uma certa classe apenas sobram 3 instâncias desta mesma classe logo não se ramos ser afetados ter os nossos resultados afetados por dados não pretendidos.

Estes pesos foram escolhidos para favorecer a escolha dos vizinhos que sejam da mesma classe.

3) [2.5v] Considering the nine training observations, learn a Bayesian classifier assuming: i) y1 and y2 are dependent; ii) { , } and { } variable sets are independent and equally important; and iii) y3 is normally distributed. Show all parameters.

3

Posterior: $P(out = class / y_1 = v_1, y_2 = v_2, y_3 = v_3)$ T. Bayes =

$$= \frac{P(out = class)\, P(y_1 = v_1, y_2 = v_2, y_3 = v_3 / out = class)}{P(y_1 = v_1, y_2 = v_2, y_3 = v_3)}$$

$$= \frac{\textcircled{1}\ P(out = class)\ \textcircled{2}\ P(y_1 = v_1, y_2 = v_2 / out = class)\ 3\ P(y_3 = v_3 / out = class)}{\underbrace{P(y_1 = v_1, y_2 = v_2)}_{\textcircled{2}'}\ \underbrace{P(y_3 = v_3)}_{3'}}$$

Dada uma nova observação O, nós sabemos classificá-la calculando $P(class / O)$ para todas as classes e N é escolher a com maior probabilidade

$$\hat{z} = \underset{c}{\operatorname{argmax}}\,(P(c/O)) = \underset{c}{\operatorname{argmax}}\ \frac{P(y_1, y_2 / c)\, P(y_3 / c)}{P(y_1, y_2)\, P(y_3)} =$$

$$= \operatorname{argmax}\ P(y_1, y_2 / c)\, P(y_3 / c)$$

igual em todos os c

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

② $P(y_1 = A, y_2 = 0 / out = P) = \frac{2}{5}$ $P(y_1 = A, y_2 = 0 / out = N) = 0$

$P(y_1 = A, y_2 = 1 / out = P) = \frac{1}{5}$ $P(y_1 = A, y_2 = 1 / out = N) = \frac{1}{4}$

$P(y_1 = B, y_2 = 0 / out = P) = \frac{1}{5}$ $P(y_1 = B, y_2 = 0 / out = N) = \frac{2}{4}$

$P(y_1 = B, y_2 = 1 / out = P) = \frac{1}{5}$ $P(y_1 = B, y_2 = 1 / out = N) = \frac{1}{4}$

③ $P(y_1 = A, y_2 = 0) = \frac{2}{9}$ $P(y_1 = B, y_2 = 0) = \frac{3}{9}$

$P(y_1 = A, y_2 = 1) = \frac{2}{9}$ $P(y_1 = B, y_2 = 1) = \frac{2}{9}$

(3)

Para $\hat{\mu}_{|out=P} = \frac{1.1+0.8+0.5+0.9+0.8}{5} = 0.82$

$\hat{\mu}_{|out=N} = \frac{1+0.9+1.2+0.9}{4} = 1$

$\sigma(y_3 | out=P) = \sqrt{\frac{(1.1-0.84)^2+(0.8-0.82)^2+(0.5-0.84)^2+(0.9-0.82)^2+(0.8-0.82)^2}{5-1}}$

$\simeq 0.2168$

$\sigma(y_3 | out=N) = \sqrt{\frac{(1-1)^2+(0.9-1)^2+(1.2-1)^2+(0.9-1)^2}{4-1}}$

$\simeq 0.1414$

$P(y_3 = x | y_{out} = class) = \frac{1}{\sqrt{2\pi} \times \sigma_{class}} \exp\left(\frac{-(x-\mu_{class})^2}{2 \times (\sigma_{class})^2}\right)$

(3')

$\hat{\mu} = \frac{1.1+0.8+0.5+0.9+0.8+1+0.9+1.2+0.9}{9}$

$= 0.9$

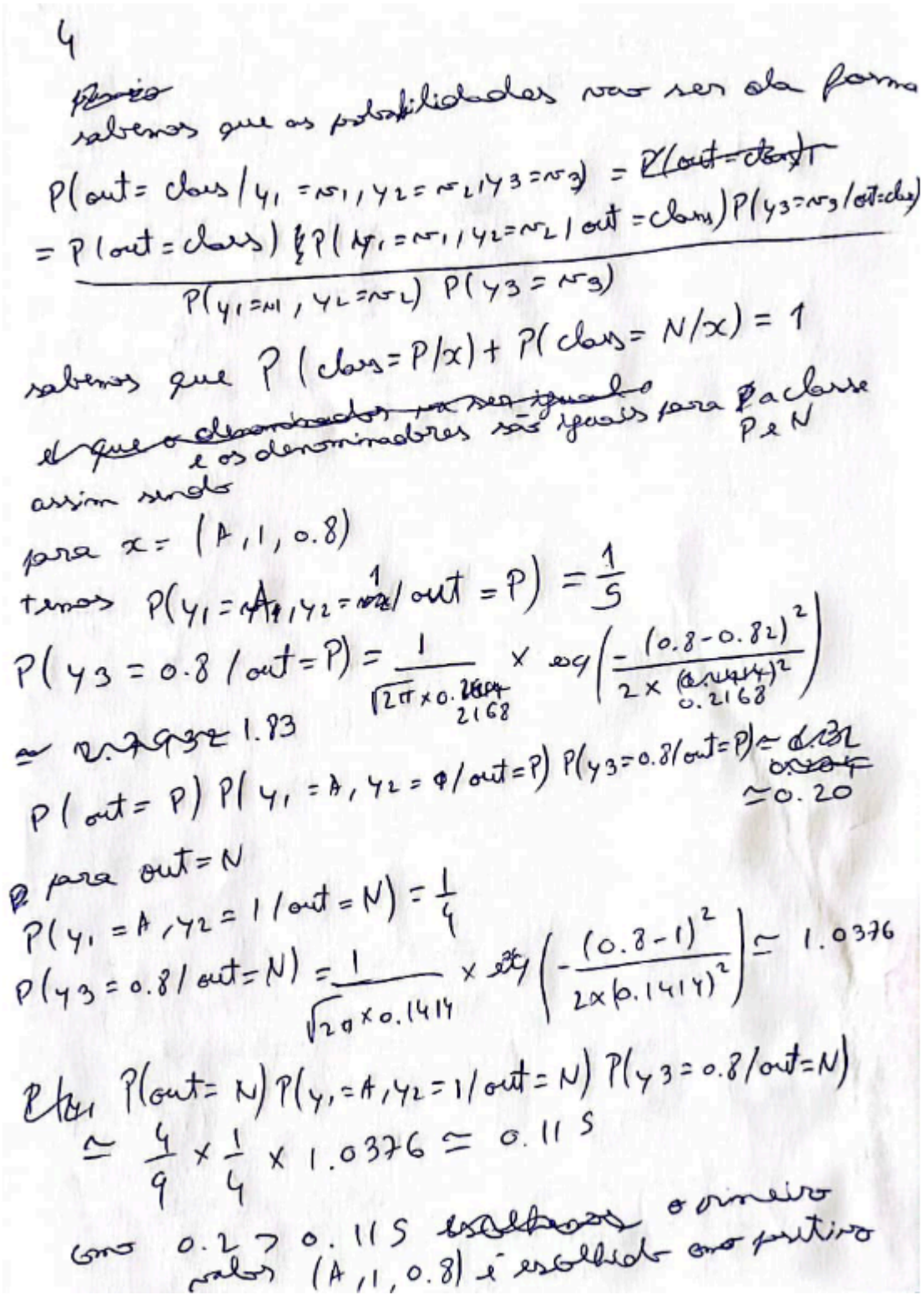
$\sigma = \sqrt{\frac{(1.1-0.9)^2+(0.8-0.9)^2+(0.5-0.9)^2+(0.9-0.9)^2+(0.8-0.9)^2+(1-0.9)^2+(0.5-0.9)^2+(1.2-0.9)^2+(0.9-0.9)^2}{9-1}}$

$= 0.2$

$P(y_3 = x) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

Já temos todos os parâmetros para aplicar o classificador

4) [2.5v] Under a MAP assumption, classify each testing observation showing all your calculus.



para (B, 1, 1)

$P(y_3 = 1 \mid out = P) = \frac{1}{\sqrt{2\pi} \times 0.2168} \times exp\left(\frac{-(1 - 0.82)^2}{2 \times (0.2168)^2}\right)$

$\simeq 1.304$

$P(out = P)\, P(y_1 = B, y_2 = 1 \mid out = P)\, P(y_3 = 1 \mid out = P)$

$= \frac{5}{9} \times \frac{1}{5} \times 1.304 \simeq 0.1449$

$P(y_3 = 1 \mid out = N) = \frac{1}{\sqrt{2\pi} \times 0.1414} \times exp\left(\frac{-(1-1)^2}{2 \times 0.1414}\right)$

$= 2.821$

$P(out = N)\, P(y_1 = B, y_2 = 1 \mid out = N)\, P(y_3 = 1 \mid out = N)$

$= \frac{4}{9} \times \frac{1}{4} \times 2.821 = 0.3134$

logo este valor é negativo

para (B, 0, 0.9)

$P(y_3 = 0.9 \mid out = P) = \frac{1}{\sqrt{2\pi} \times 0.2168} \times exp\left(\frac{-(0.9 - 0.82)^2}{2 \times (0.2168)^2}\right)$

$\simeq 1.719$

$P(out = P)\, P(y_1 = B, y_2 = 0 \mid out = P)\, P(y_3 = 0.9 \mid out = P) \simeq$

$\simeq \frac{5}{9} \times 1.719 \times \frac{1}{5} \simeq 0.191$

$P(y_3 = 0.9 \mid out \neq N) = \frac{1}{\sqrt{2\pi} \times 0.1414} \times exp\left(\frac{-(0.9 - 1)^2}{2 \times (0.1414)^2}\right)$

$\simeq 2.197$

$P(out = N)\, P(y_1 = B, y_2 = 0 \mid out = N)\, P(y_3 = 0.9 \mid out = N) =$

$= 2.197 \times \frac{4}{9} \times \frac{2}{4} = 0.488$ assim sendo

$0.191 < 0.488$ logo

(B; 0; 0.9) é negativo

5) [2.5v] Using a naïve Bayes under a ML assumption, classify the new sentence "*I like to run*" . For the likelihoods calculation consider the following formula,

$$p(ti|c) = (freq(ti) + 1)/(Nc + V) ,$$

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

(5)

Dos Positivos — termos unicos: Amazing rum I like it

Amazing rum
I like it

NP = 5

Negativos — termos unicos: too tired Bad rum

too tired.
Bad rum

$N_N = 4$

total de termos = 8 = V

usar a formula para calcular a probabilidade de cada termo

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

para P

$P("I"/out=P) = \frac{1+1}{5+8} = \frac{2}{13}$

$P("like"/out=P) = \frac{1+1}{5+8} = \frac{2}{13}$

$P("to"/out=P) = \frac{1+0}{5+8} = \frac{1}{13}$

$P("rum"/out=P) = \frac{1+1}{5+8} = \frac{2}{13}$

para N

$P("I"/out=N) = \frac{0+1}{4+8} = \frac{1}{12}$

$P("like"/out=N) = \frac{0+1}{4+8} = \frac{1}{12}$

$P("to"/out=N) = \frac{0+1}{4+8} = \frac{1}{12}$

$P("rum"/out=N) = \frac{1+1}{4+8} = \frac{2}{12}$

TÉCNICO LISBOA

Aprendizagem 2022/23

Homework II – Group 51

(ist1105940, ist1106221)

$P("I\ like\ to\ run"/out=P) = P("I"/out=P) * P("like"/out=P) * P("to"/out=P) * P("run"/out=P) =$

$= \frac{2}{13} \times \frac{2}{13} \times \frac{1}{13} \times \frac{2}{13} = \frac{8}{28561} = 2.8 \times 10^{-4}$

$P("I\ like\ to\ run"/out=N) = P("I"/out=N) \times P("like"/out=N) \times P("to"/out=N) \times P("run"/out=N) = \frac{1}{12} \times \frac{1}{12} \times \frac{1}{12} \times \frac{2}{12} = \frac{2}{20736}$

$\simeq 9.654 \times 10^{-5}$

$P("I\ like\ to\ run"/P) > P("I\ like\ to\ run"/N)$

a frase é classificada como Positivo

para (P)

argmax $P(A) \prod_{i=1}^{n} P(A_i = x_j)$

k

Como ambos têm o mesmo Prior basta apenas comparar as probabilidades obtidas anteriores e com isso podemos afirmar que a frase é classificada como positiva $P(P) = P(N) = 0.5$

II. Programming and critical analysis

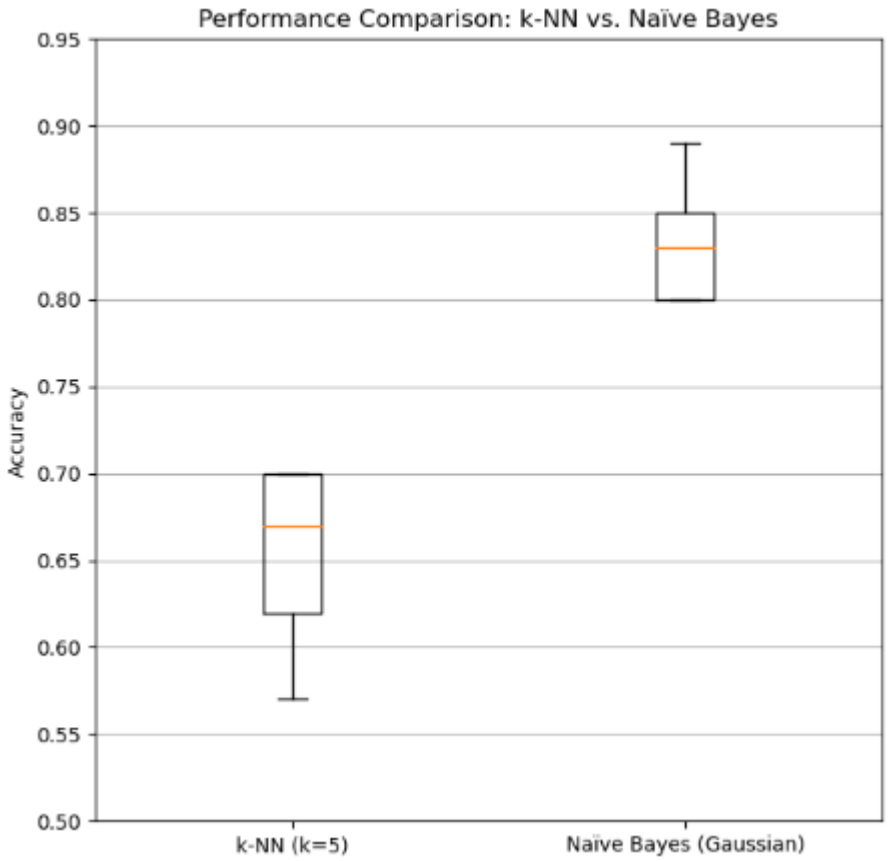
1) Compare the performance of a kNN $k = 5$ with and a naïve Bayes with Gaussian assumption (consider all remaining parameters as default):

a) [1.0v] 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 pandas as pd
from sklearn.model_selection import StratifiedKFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
import matplotlib.pyplot as plt

# Reading the CSV file
df = pd.read_csv('heart-disease.csv')

X = df.drop('target', axis=1)
y = df['target']

knn_predictor = KNeighborsClassifier(n_neighbors=5)
nb_predictor = GaussianNB()

folds = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)
knn_accuracies = []
nb_accuracies = []


# iterate per fold
for train_k, test_k in folds.split(X, y):
    X_train, X_test = X.iloc[train_k], X.iloc[test_k]
    y_train, y_test = y.iloc[train_k], y.iloc[test_k]

    # train and assess k-NN
    knn_predictor.fit(X_train, y_train)
    y_pred = knn_predictor.predict(X_test)
    knn_accuracies.append(round(metrics.accuracy_score(y_test, y_pred),2))

    # train and assess Naïve Bayes with Gaussian assumption
    nb_predictor.fit(X_train, y_train)
    y_pred = nb_predictor.predict(X_test)
    nb_accuracies.append(round(metrics.accuracy_score(y_test, y_pred),2))


# plots
plt.figure(figsize=(7, 7))
plt.boxplot([knn_accuracies, nb_accuracies], labels=['k-NN (k=5)', 'Naïve Bayes (Gaussian)'])
plt.title('Performance Comparison: k-NN vs. Naïve Bayes')
plt.ylabel('Accuracy')
plt.ylim(0.5, 0.95)
plt.grid(axis='y')
plt.show()
```

Homework II – Group 51

(ist1105940, ist1106221)



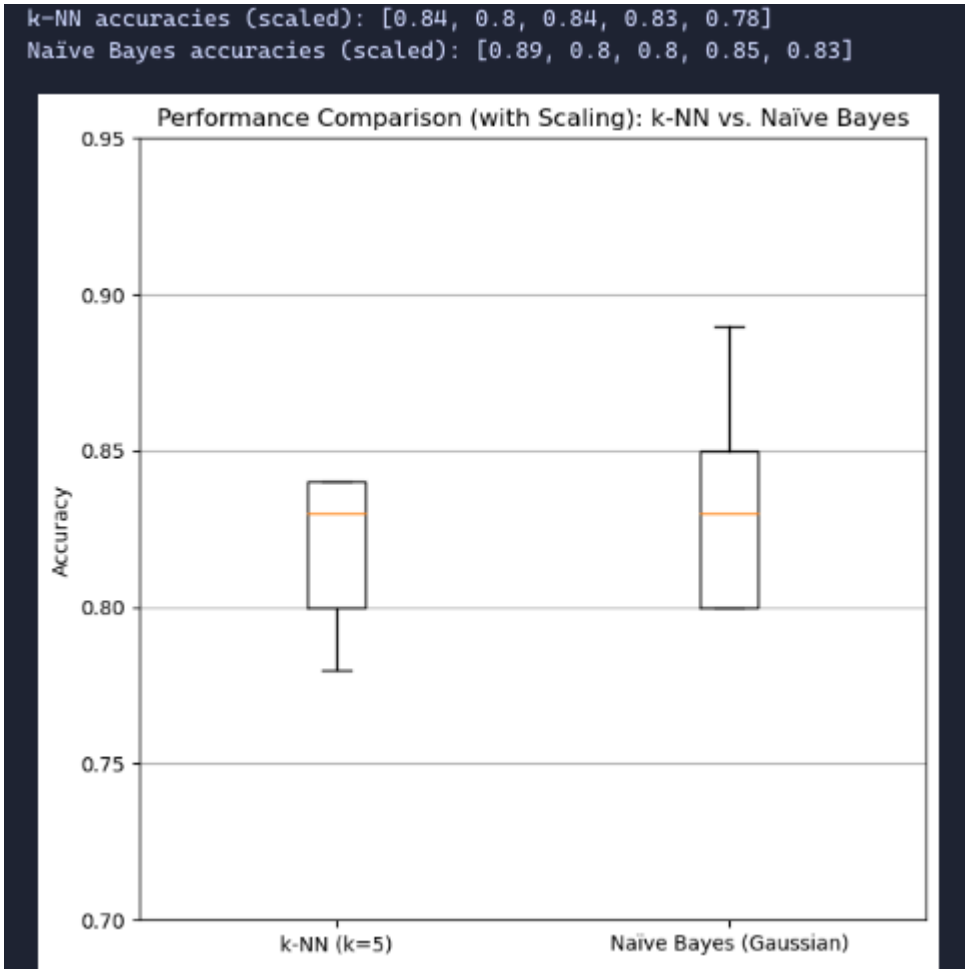
O modelo Naïve Bayes tem consistentemente maior precisão em comparação com o modelo k-NN. O modelo k-NN exibe maior variabilidade no desempenho, como indicado pela maior dispersão no boxplot. O modelo Naïve Bayes é mais estável em termos de desempenho, provavelmente devido à sua natureza probabilística, enquanto a precisão do k-NN depende mais da estrutura local dos dados e é mais sensível ao tamanho e distribuição dos vizinhos.

b) [1.0v] 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 pandas as pd
from sklearn.model_selection import StratifiedKFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
import matplotlib.pyplot as plt

# Reading the CSV file
df = pd.read_csv('heart-disease.csv')

X = df.drop('target', axis=1)
y = df['target']

# Initializing the models
knn_predictor = KNeighborsClassifier(n_neighbors=5)
nb_predictor = GaussianNB()

# Stratified k-folds
folds = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)
knn_accuracies = []
nb_accuracies = []

# Min-Max Scaler
scaler = MinMaxScaler()

# iterate per fold
for train_k, test_k in folds.split(X, y):
    X_train, X_test = X.iloc[train_k], X.iloc[test_k]
    y_train, y_test = y.iloc[train_k], y.iloc[test_k]

    # Scaling the data using Min-Max Scaler
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    # Train and assess k-NN
    knn_predictor.fit(X_train_scaled, y_train)
    y_pred_knn = knn_predictor.predict(X_test_scaled)
    knn_accuracies.append(round(metrics.accuracy_score(y_test, y_pred_knn), 2))

    # Train and assess Naïve Bayes with Gaussian assumption
    nb_predictor.fit(X_train_scaled, y_train)
    y_pred_nb = nb_predictor.predict(X_test_scaled)
    nb_accuracies.append(round(metrics.accuracy_score(y_test, y_pred_nb), 2))

# Report the accuracy
print("k-NN accuracies (scaled):", knn_accuracies)
print("Naïve Bayes accuracies (scaled):", nb_accuracies)

# Boxplot to compare
plt.figure(figsize=(7, 7))
plt.boxplot([knn_accuracies, nb_accuracies], labels=['k-NN (k=5)', 'Naïve Bayes (Gaussian)'])
plt.title('Performance Comparison (with Scaling): k-NN vs. Naïve Bayes')
plt.ylabel('Accuracy')
plt.ylim(0.7, 0.95)
plt.grid(axis='y')
plt.show()
```

Homework II – Group 51

(ist1105940, ist1106221)



O desempenho do k-NN melhorou significativamente após a escala, mostrando precisões mais altas e consistentes em todos os folds. O desempenho do Naïve Bayes permaneceu relativamente estável, sem mudanças perceptíveis após a escala. Isso indica que o k-NN é mais sensível à escala dos dados, provavelmente porque depende de métricas de distância, enquanto o Naïve Bayes não depende de distância e, portanto, é menos afetado pela escala das características.

c) [1.0v] Using scipy, test the hypothesis “the *kNN* model is statistically superior to naïve Bayes regarding accuracy”, asserting whether it is true

```python
import pandas as pd
from sklearn.model_selection import StratifiedKFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from scipy import stats

# Reading the CSV file
df = pd.read_csv('heart-disease.csv')

X = df.drop('target', axis=1)
y = df['target']

knn_predictor = KNeighborsClassifier(n_neighbors=5)
nb_predictor = GaussianNB()

# 10-fold stratified cross-validator with shuffling
folds = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)

knn_accuracies = []
nb_accuracies = []

# iterate per fold
for train_k, test_k in folds.split(X, y):
    X_train, X_test = X.iloc[train_k], X.iloc[test_k]
    y_train, y_test = y.iloc[train_k], y.iloc[test_k]

    # train and assess k-NN
    knn_predictor.fit(X_train, y_train)
    y_pred = knn_predictor.predict(X_test)
    knn_accuracies.append(round(metrics.accuracy_score(y_test, y_pred),2))

    # train and assess Naïve Bayes with Gaussian assumption
    nb_predictor.fit(X_train, y_train)
    y_pred = nb_predictor.predict(X_test)
    nb_accuracies.append(round(metrics.accuracy_score(y_test, y_pred),2))


# kNN is better than naïve Bayes?
res = stats.ttest_rel(knn_accuracies, nb_accuracies, alternative='greater')
print("kNN > naïve Bayes? pval=",res.pvalue)
```

✓ 0.1s Python

```
kNN > naïve Bayes? pval= 0.998415501126768
```

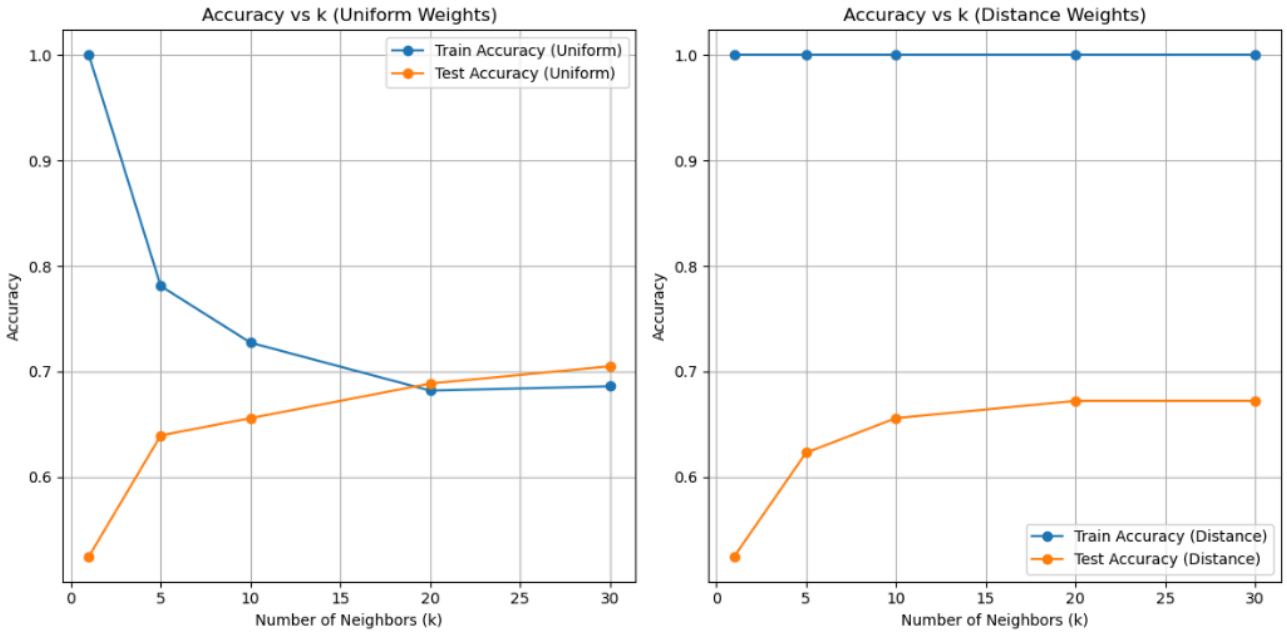
kNN > naïve Bayes? pval= 0.998415501126768 como pval > valores de significancia usuais (ou seja, pval é superior a 0.01, 0.05 e 0.1), não é possível rejeitar a hipótese nula e afirmar que o kNN é estatisticamente superior ao Naive Bayes.

2) Using a 80-20 train-test split, vary the number of neighbors of a kNN classifier using weights k = {1, 5, 10, 20, 30} . Additionally, for each k , train one classifier using uniform and distance weights.

a. [1.0v] Plot the train and test accuracy for each model.

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

# Reading the CSV file
df = pd.read_csv('heart-disease.csv')

X = df.drop('target', axis=1)
y = df['target']

# Split the data into 80% train and 20% test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)


# List of k values to try
k_values = [1, 5, 10, 20, 30]

# Store accuracies
train_accuracies_uniform = []
test_accuracies_uniform = []
train_accuracies_distance = []
test_accuracies_distance = []

# Iterate over each k value
for k in k_values:
    # Uniform weights
    knn_uniform = KNeighborsClassifier(n_neighbors=k, weights='uniform')
    knn_uniform.fit(X_train, y_train)
    train_acc_uniform = metrics.accuracy_score(y_train, knn_uniform.predict(X_train))
    test_acc_uniform = metrics.accuracy_score(y_test, knn_uniform.predict(X_test))
    train_accuracies_uniform.append(train_acc_uniform)
    test_accuracies_uniform.append(test_acc_uniform)

    # Distance weights
    knn_distance = KNeighborsClassifier(n_neighbors=k, weights='distance')
    knn_distance.fit(X_train, y_train)
    train_acc_distance = metrics.accuracy_score(y_train, knn_distance.predict(X_train))
    test_acc_distance = metrics.accuracy_score(y_test, knn_distance.predict(X_test))
    train_accuracies_distance.append(train_acc_distance)
    test_accuracies_distance.append(test_acc_distance)

# Plot the results
plt.figure(figsize=(12, 6))
```

```python
# Uniform weights plot
plt.subplot(1, 2, 1)
plt.plot(k_values, train_accuracies_uniform, marker='o', label='Train Accuracy (Uniform)')
plt.plot(k_values, test_accuracies_uniform, marker='o', label='Test Accuracy (Uniform)')
plt.title('Accuracy vs k (Uniform Weights)')
plt.xlabel('Number of Neighbors (k)')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)

# Distance weights plot
plt.subplot(1, 2, 2)
plt.plot(k_values, train_accuracies_distance, marker='o', label='Train Accuracy (Distance)')
plt.plot(k_values, test_accuracies_distance, marker='o', label='Test Accuracy (Distance)')
plt.title('Accuracy vs k (Distance Weights)')
plt.xlabel('Number of Neighbors (k)')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)

plt.tight_layout()
plt.show()
```



b. [1.5v] Explain the impact of increasing the neighbors on the generalization ability of the models.

Pesos Uniformes (Gráfico 1 à Esquerda):

k=1: O modelo apresenta uma acurácia de treino perfeita, mas a acurácia de teste é baixa. Este é um claro indicativo de overfitting, onde o modelo está a memorizar os dados de treino, mas falha em generalizar para dados de teste.

k=5 a k=10: Observa-se uma melhoria na acurácia de teste.

k>10: À medida que k aumenta, a acurácia de treino vai aumentando ultrapassando a acurácia de treino em k=20. Esta diminuição da acurácia de treino e aumento lento da acurácia de teste pode sugerir uma leve tendência de underfitting, onde, com muitos vizinhos, o modelo perde a capacidade de capturar variações locais nos dados.

Pesos Baseados na Distância (Gráfico 2 à Direita):

k=1: O modelo novamente apresenta acurácia de treino perfeita, mas a acurácia de teste permanece baixa, semelhante ao caso anterior.

k=5 a k=10: Para valores intermediários de k, especialmente até de k=10,a acuracia de teste é semelhante ao modelo com pesos uniformes.

k>10: Com k maior que 10, a acurácia de teste estabiliza. Embora a acurácia de treino continue constante em 1.0, o impacto de atribuir pesos mais altos aos vizinhos mais próximos ajuda a mitigar a queda na acurácia de teste, mantendo um grau de localidade nas decisões.

Impacto no Poder de Generalização: Aumentar o número de vizinhos tende a suavizar o modelo, reduzindo o overfitting e melhorando a generalização até certo ponto. Contudo, quando k é excessivamente grande, pode resultar em underfitting, levando a uma queda na acurácia tanto de treino quanto de teste(neste caso não acontece redução da acurácia de treino).

O uso de pesos baseados na distância atenua o impacto negativo de um aumento excessivo de vizinhos, já que o modelo ainda prioriza os vizinhos mais próximos, preservando um nível de localidade nas suas decisões.

Resumindo, aumentar k ajuda a reduzir o overfitting, mas um valor muito elevado pode resultar em underfitting. O uso de pesos por distância geralmente melhora a generalização para valores maiores de k.

3) [1.5v] 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.

O Naïve Bayes supõem que todas as características (features) são independentes entre si, para uma dada classe alvo. Porém, no conjunto de dados de doenças cardíacas, é provável que muitas das características sejam correlacionadas. Por exemplo, variáveis como nível de colesterol, pressão arterial e idade podem ter dependências entre si, o que viola essa suposição de independência. Isto pode penalizar o desempenho do modelo Naïve Bayes, que não consegue modelar essas correlações entre os features.

Este modelo assume que os dados numéricos seguem uma distribuição normal (gaussiana). Se as features numéricas do conjunto de dados de doenças cardíacas não seguirem a distribuição normal (forem muito assimétricos, ou tiverem outras distribuições), o Naïve Bayes pode não se ajustar bem aos dados. Dados como níveis de colesterol ou idade podem não seguir uma distribuição normal, podendo diminuir a capacidade do modelo de fazer previsões precisas.

END