Aprendizagem 2021/22

Homework II – Group 105

I. Pen-and-paper

1) Answer 1

	x1	x2	x3	x4	x5	x6	x7	x8
x1	-----------	5/2	3/2	1/2	3/2	3/2	3/2	5/2
x2	5/2	-----------	3/2	5/2	3/2	3/2	3/2	1/2
x3	3/2	3/2	-----------	3/2	5/2	5/2	1/2	3/2
x4	1/2	5/2	3/2	-----------	3/2	3/2	3/2	5/2
x5	3/2	3/2	5/2	3/2	-----------	1/2	5/2	3/2
x6	3/2	3/2	5/2	3/2	1/2	-----------	5/2	3/2
x7	3/2	3/2	1/2	3/2	5/2	5/2	-----------	3/2
x8	5/2	1/2	3/2	5/2	3/2	3/2	3/2	-----------

Cálculo da distância para cada observação e seleção (a verde) dos respetivos vizinhos.

Moda ponderada:

x1: ((2/3 + 2/1)*P, (2/3 + 2/3 + 2/3)*N) = P > N => TP

x2: ((2/3)*P, (2/3 + 2/3 + 2/3 + 2/1)*N) = N > P => FN

x3: ((2/3 + 2/3)*P, (2/3 + 2/1 + 2/3)*N) = N > P => FN

x4: ((2/1 + 2/3)*P, (2/3 + 2/3 + 2/3)*N) = P > N => TP

x5: ((2/3 + 2/3 + 2/3)*P, (2/1 + 2/3)*N) = N > P => TN

x6: ((2/3 + 2/3 + 2/3)*P, (2/1 + 2/3)*N) = N > P => TN

x7: ((2/3 + 2/3 + 2/1 + 2/3)*P, (2/3)*N) = P > N => FP

x8: ((2/1 + 2/3)*P, (2/3 + 2/3 + 2/3)*N) = P > N => FP

		true	
		P	N
prediction	P	TP 2	FP 2
	N	FN 2	TN 2

Aprendizagem 2021/22
Homework II – Group 105

Recall = TP/(TP + FN) = 2/(2+2) = ½

	y_1	y_2	class
x_1	A	0	P
x_2	B	1	P
x_3	A	1	P
x_4	A	0	P
x_5	B	0	N
x_6	B	0	N
x_7	A	1	N
x_8	B	1	N

Aprendizagem 2021/22
Homework II – Group 105

2)

	y_1	y_2	y_3	Class
x_1	A	0	1.2	1
x_2	B	1	0.8	1
x_3	A	1	0.5	1
x_4	A	0	0.9	1
x_5	B	0	1	0
x_6	B	0	0.9	0
x_7	A	1	1.2	0
x_8	B	1	0.8	0
x_9	B	0	0.8	1

$1 \rightarrow$ positive
$0 \rightarrow$ negative

class = z

$p(y_1 = A, y_2 = 0 \mid z = 1) = \frac{2}{5}$
$p(y_1 = A, y_2 = 1 \mid z = 1) = \frac{1}{5}$
$p(y_1 = B, y_2 = 0 \mid z = 1) = \frac{1}{5}$
$p(y_1 = B, y_2 = 1 \mid z = 1) = \frac{1}{5}$

$p(y_1 = A, y_2 = 0 \mid z = 0) = 0$
$p(y_1 = A, y_2 = 1 \mid z = 0) = \frac{1}{4}$
$p(y_1 = B, y_2 = 0 \mid z = 0) = \frac{2}{4}$
$p(y_1 = B, y_2 = 1 \mid z = 0) = \frac{1}{4}$

$p(y_3 \mid z = 0)$

$$\mu = \frac{1 + 0.9 + 1.2 + 0.8}{4} = 0.975$$

$$\sigma^2 = \frac{1}{3} \sum_{1}^{4} (y_{3_i} - \mu)^2$$
$$= \frac{1}{3} \times \left[(1 - 0.975)^2 + (0.9 - 0.975)^2 + (1.2 - 0.975)^2 + (0.8 - 0.975)^2 \right]$$
$$= 0.029167$$

$$p(y_3 \mid z = 0) = \frac{1}{\sqrt{2\pi \times 0.029167}} \times e^{-\frac{1}{2 \times 0.029167} \times (y_3 - 0.975)^2}$$

$$p(x) = p(x \mid z=0)\,p(z=0) + p(x \mid z=1)\,p(z=1)$$
$$= p(y_1, y_2 \mid z=0)\,p(y_3 \mid z=0)\,p(z=0) + p(y_1, y_2 \mid z=1)\,p(y_3 \mid z=1)\,p(z=1)$$

$$p(z=0) = \frac{4}{9}$$
$$p(z=1) = \frac{5}{9}$$
$$p(z=0 \mid x) = 1 - p(z=1 \mid x)$$

$$p(z=1 \mid x) = \frac{p(x \mid z=1)\,p(z=1)}{p(x)} = \frac{p(x \mid z=1)\,p(z=1)}{p(x_{y_1}, x_{y_2} \mid z=0)\,p(x_{y_3} \mid z=0)\,p(z=0) + p(x_{y_1}, x_{y_2} \mid z=1)\,p(x_{y_3} \mid z=1)\,p(z=1)}$$

$p(y_3 \mid z=1)$

$$\mu = \frac{1.2 + 0.8 + 0.5 + 0.9 + 0.8}{5} = 0.84$$

$$\sigma^2 = \frac{1}{4} \sum_{1}^{5} (y_{3_i} - \mu)^2$$
$$= \frac{1}{4} \times \left[(1.2 - 0.84)^2 + (0.8 - 0.84)^2 + (0.5 - 0.84)^2 + (0.9 - 0.84)^2 + (0.8 - 0.84)^2 \right]$$
$$= 0.063$$

$$p(y_3 \mid z=1) = \frac{1}{\sqrt{2\pi \times 0.063}} \times e^{-\frac{1}{2 \times 0.063} \times (y_3 - 0.84)^2}$$

3)

	y_1	y_2	y_3	class
x_1	A	1	0.8	1
x_2	B	1	1	1
x_3	B	0	0.9	0

positive = 1
negative = 0

para x_1:

$$p(y_3 = 0.8 \mid z=0) = \frac{1}{\sqrt{2\pi \times 0.029167}} \times e^{-\frac{1}{2\times 0.029167} \times (0.8-0.975)^2} = 1.38185$$

$$p(y_3 = 0.8 \mid z=1) = \frac{1}{\sqrt{2\pi \times 0.063}} \times e^{-\frac{1}{2\times 0.063} \times (0.8-0.84)^2} = 1.56937$$

$$p(x_1 \mid z=1) = p(y_1 = A, y_2 = 1 \mid z=1)\, p(y_3 = 0.8 \mid z=1) = \frac{1}{5} \times 1.56937 = 0.313874$$

$$p(z=1 \mid x_1) = \frac{p(x \mid z=1)\,p(z=1)}{P(X_{y_1}, X_{y_2} \mid z=0)\,p(X_{y_3} \mid z=0)\,p(z=0) + p(X_{y_1}, X_{y_2} \mid z=1)\,p(X_{y_3} \mid z=1)\,p(z=1)}$$

$$= \frac{0.313874 \times \frac{5}{9}}{\frac{1}{4} \times 1.38185 \times \frac{4}{9} + \frac{1}{5} \times 1.56937 \times \frac{5}{9}}$$

$$= \frac{0.174374}{0.153539 + 0.174374} = 0.5317690973$$

para x_2:

$$p(y_3 = 1 \mid z=0) = \frac{1}{\sqrt{2\pi \times 0.029167}} \times e^{-\frac{1}{2\times 0.029167} \times (1-0.975)^2} = 2.31106$$

$$p(y_3 = 1 \mid z=1) = \frac{1}{\sqrt{2\pi \times 0.063}} \times e^{-\frac{1}{2\times 0.063} \times (1-0.84)^2} = 1.29719$$

$$p(x_2|z=1) = p(y_1=B, y_2=1|z=1)\,p(y_3=1|z=1) = \frac{1}{5} \times 1.29719 = 0.259438$$

$$p(z=1|x_2) = \frac{p(x|z=1)p(z=1)}{P(x_{y_1},x_{y_2}|z=0)p(x_{y_3}|z=0)p(z=0) + p(x_{y_1},x_{y_2}|z=1)p(x_{y_3}|z=1)p(z=1)}$$

$$= \frac{0.259438 \times 5/9}{\frac{1}{4} \times 2.31106 \times \frac{4}{9} + \frac{1}{5} \times 1.29719 \times \frac{5}{9}}$$

$$= \frac{0.144132}{0.256784 + 0.144132} = 0.359507$$

para x_3 :

$$p(y_3=0.9|z=0) = \frac{1}{\sqrt{2\pi \times 0.029167}} \times e^{-\frac{1}{2\times 0.029167} \times (0.9-0.975)^2} = 2.12122$$

$$p(y_3=0.9|z=1) = \frac{1}{\sqrt{2\pi \times 0.063}} \times e^{-\frac{1}{2\times 0.063} \times (0.9-0.84)^2} = 1.54465$$

$$p(x_3|z=1) = p(y_1=B, y_2=0|z=1)\,p(y_3=0.9|z=1) = \frac{1}{5} \times 1.54465 = 0.30893$$

$$p(z=1|x_3) = \frac{p(x|z=1)p(z=1)}{P(x_{y_1},x_{y_2}|z=0)p(x_{y_3}|z=0)p(z=0) + p(x_{y_1},x_{y_2}|z=1)p(x_{y_3}|z=1)p(z=1)}$$

$$= \frac{0.30893 \times 5/9}{\frac{2}{4} \times 2.12122 \times \frac{4}{9} + \frac{1}{5} \times 1.54465 \times \frac{5}{9}}$$

$$= \frac{0.171628}{0.471382 + 0.171628} = 0.266913$$

Utilizando o MAP

$$p(h_n|D) = \frac{p(D|h_n) \cdot p(h_n)}{p(D)}$$

em que D equivale a ser positivo, h_n a uma observação e $h_{MAP} = \arg\max_{h \in H} p(h_n|D)$,

podemos concluir que a observação $x_1 = \left(\begin{pmatrix} A \\ 1 \\ 0.8 \end{pmatrix}, \text{Positive}\right)$ é a que tem maior probabilidade

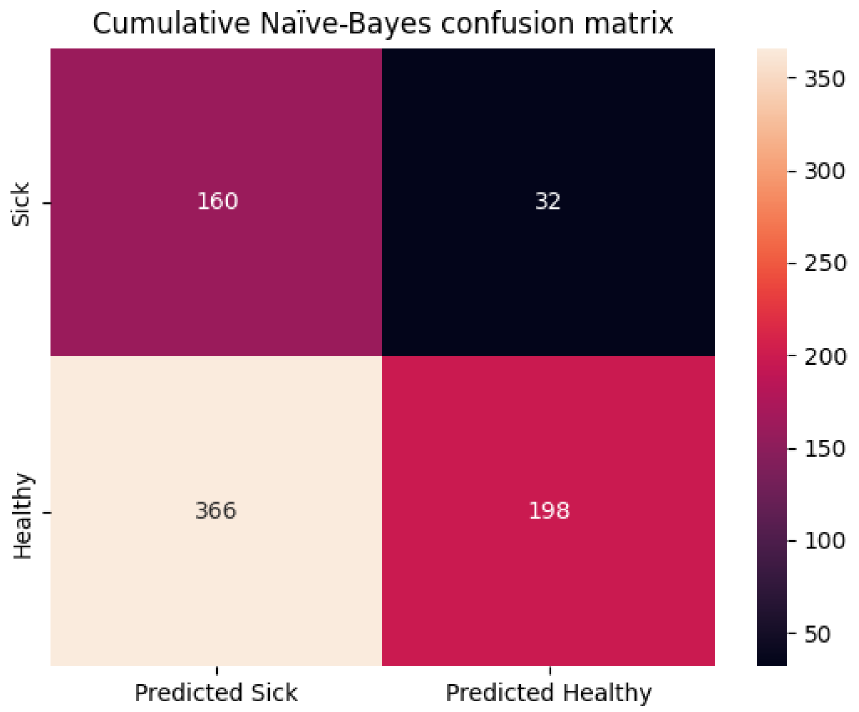
4) Apesar tamanho da amostra ser relativamente pequeno, o threshold que nos garante maior precisão é o de 0.3 e, portanto, é o que deve ser considerado.

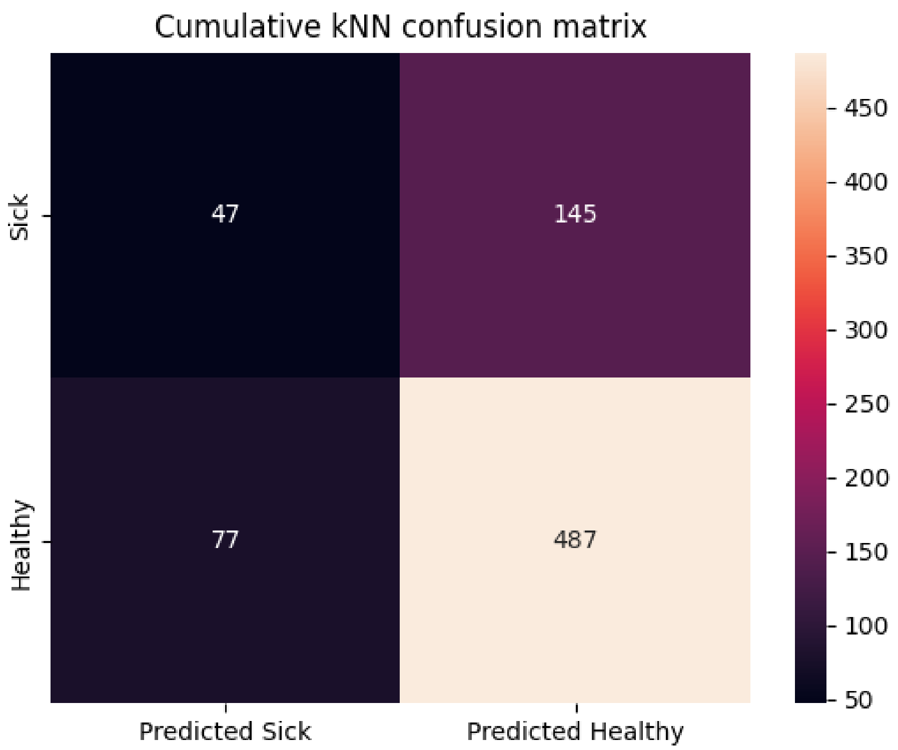
	P(Z=1|X)	0.3	0.5	0.7	Real
X1	0.531769	1	1	0	1
X2	0.359507	1	0	0	1
X3	0.266913	0	0	0	0
	Accuracy	3/3	2/3	1/3	

Aprendizagem 2021/22
Homework II – Group 105

II. Programming and critical analysis

5)





6) Precisão Naïve-Bayes: 0.5 ± 0.12

Precisão kNN: 0.69 ± 0.06

Naïve-Bayes > kNN? pval= 0.9999932386615072

Naïve-Bayes < kNN? pval= 6.7613384927759316e-06

Naïve-Bayes != kNN? pval= 1.3522676985551863e-05

Através do valor de precisão e do facto do valor-p ser inferior a 0.05 nas segunda e terceira hipóteses, podemos aferir que o modelo kNN é significativamente melhor estatisticamente que o modelo Naïve-Bayes para este dataset.

7) O modelo kNN varia acentuadamente consoante o número e o peso dos vizinhos. Neste caso, o nosso dataset provou ter vizinhos valiosos, permitindo assim a este modelo ter uma grande precisão. Adicionalmente, o modelo Naïve-Bayes interpreta todas as variáveis como independentes, e neste caso, como existem dependências entre as variáveis, prejudicou a sua performance. Além disso, como podemos verificar nos valores de precisão abaixo, o Naïve-Bayes poderá ter sofrido underfitting, devido à baixa precisão tanto de treino como de teste.

Overfit ou underfit NB?

- Training accuracy: 0.49
- Testing accuracy: 0.47

III. APPENDIX

```
import warnings
from sklearn import metrics, datasets, tree
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from scipy import stats
from scipy.io.arff import loadarff
from sklearn.naive_bayes import GaussianNB
import seaborn as sns
from sklearn.preprocessing import normalize

def warn(*args, **kwargs):
    pass
warnings.warn = warn

# Reading the ARFF file
data = loadarff('pd_speech.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')
X, y = df[list(df.columns[:-1])], df[["class"]]

X = normalize(X)
X = pd.DataFrame(X)

predictor_NB = GaussianNB()
predictor_kNN = KNeighborsClassifier(n_neighbors=5, p=2, weights="uniform")

folds_acc_NB = []
folds_acc_kNN = []
overfit_NB = []
overfit_kNN = []
```

```
total_confusion_NB = np.array(((0, 0), (0, 0)))
total_confusion_kNN = np.array(((0, 0), (0, 0)))
folds = StratifiedKFold(n_splits=10, random_state=0, shuffle=True)

# 0 = Sick; 1 = Healthy
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]

    predictor_NB.fit(X_train, y_train)
    y_pred_NB = predictor_NB.predict(X_test)
    cm_NB = np.array(confusion_matrix(y_test, y_pred_NB, labels=["0", "1"]))
    folds_acc_NB.append(round(metrics.accuracy_score(y_test, y_pred_NB), 2))
    y_pred_NB = predictor_NB.predict(X_train)
    overfit_NB.append(round(metrics.accuracy_score(y_train, y_pred_NB), 2))
    total_confusion_NB = np.add(total_confusion_NB, cm_NB)

    predictor_kNN.fit(X_train, y_train)
    y_pred_kNN = predictor_kNN.predict(X_test)
    cm_kNN = np.array(confusion_matrix(y_test, y_pred_kNN, labels=["0", "1"]))
    folds_acc_kNN.append(round(metrics.accuracy_score(y_test, y_pred_kNN), 2))
    y_pred_kNN = predictor_kNN.predict(X_train)
    overfit_kNN.append(round(metrics.accuracy_score(y_train, y_pred_kNN), 2))
    total_confusion_kNN = np.add(total_confusion_kNN, cm_kNN)

confusion_NB = pd.DataFrame(total_confusion_NB, index=["Sick", "Healthy"], columns=["Predicted
Sick", "Predicted Healthy"])
confusion_kNN = pd.DataFrame(total_confusion_kNN, index=["Sick", "Healthy"], columns=["Predicted
Sick", "Predicted Healthy"])

heat = sns.heatmap(confusion_NB, annot=True, fmt='g')
plt.title("Cumulative Naïve-Bayes confusion matrix")
plt.show()
heat2 = sns.heatmap(confusion_kNN, annot=True, fmt='g')
plt.title("Cumulative kNN confusion matrix")
plt.show()

classifiers = (
    ("Naive Bayes", predictor_NB),
    ("kNN", predictor_kNN)
)

print("Overfit NB?\nTraining accuracy:", round(sum(overfit_NB)/len(overfit_NB), 2), "\nTesting
accuracy:", round(sum(folds_acc_NB)/len(folds_acc_NB), 2))
print("Overfit kNN?\nTraining accuracy:", round(sum(overfit_kNN)/len(overfit_kNN), 2), "\nTesting
accuracy:", round(sum(folds_acc_kNN)/len(folds_acc_kNN), 2))

for name, classifier in classifiers:
    accs = cross_val_score(classifier, X, y, cv=10, scoring='accuracy')
    print(name, "accuracy =", round(np.mean(accs), 2), "±", round(np.std(accs), 2))

# NB > kNN?
res = stats.ttest_rel(folds_acc_NB, folds_acc_kNN, alternative='greater')
print("p1>p2? pval=", res.pvalue)
# NB < kNN?
res = stats.ttest_rel(folds_acc_NB, folds_acc_kNN, alternative='less')
print("p1<p2? pval=", res.pvalue)
# NB != kNN?
res = stats.ttest_rel(folds_acc_NB, folds_acc_kNN, alternative='two-sided')
print("p1!=p2? pval=", res.pvalue)
```

END