Aprendizagem 2022/23

Homework II

Deadline 17/10/2022 (Monday) 23:59 via Fenix as PDF

I. Pen-and-paper [13v]

1)

	x1	x2	x3	x4	x5	x6	x7	x8
x1	–	2.5	1.5	0.5	1.5	1.5	1.5	2.5
x2		–	1.5	2.5	1.5	1.5	1.5	0.5
x3			–	1.5	2.5	2.5	0.5	1.5
x4				–	1.5	1.5	1.5	0.5
x5					–	0.5	2.5	1.5
x6						–	2.5	1.5
x7							–	1.5
x8								–

$w_{i,j} = \frac{1}{d(xi, xj)}$

5NN(x1) = (x3, x4, x5, x6, x7) = argmax(($\frac{2}{3}$ + 2)P, (3*$\frac{2}{3}$)N) = P

5NN(x2) = (x3, x5, x6, x7, x8) = argmax($\frac{2}{3}$P, (3x$\frac{2}{3}$+2)N) = N

5NN(x3) = ………. = N
5NN(x4) = ………. = P
5NN(x5) = ………. = N
5NN(x6) = ………. = N
5NN(x7) = ………. = P
5NN(x8) = ………. = P

Reais \ Previstos	P	N
P	2	2
N	2	2

$Recall_P = \frac{TP}{TP + FN} = \frac{2}{2+2} = 0.5 = Recall_N\ (porque\ os\ valores\ são\ todos\ iguais)$

2)A=0, B=1

Para z = P, y1y2{

$\mu = \frac{1}{5}([\frac{0}{0}] + [\frac{1}{1}] + [\frac{0}{1}] + [\frac{0}{0}] + [\frac{1}{0}]) = [\frac{0.4}{0.4}]$ (not a fraction, can't do "vertical" vector)

var(y1) = $\frac{1}{4}\sum_{i=1}^{5}(y_{1i} - \mu)^2 = ... = 0.1812$; var(y2) = … = 0.1812

cov(y1, y2)= $\frac{1}{4}\sum_{i=1}^{5}(y_{1i} - \mu)(y_{2i} - \mu) = ... = 0.1098$

$\Sigma = [\frac{0.1812 \quad 0.1098}{0.1098 \quad 0.1812}]$; $|\Sigma| = 0.02077$; $\Sigma^{-1} = \frac{1}{0.02077}\Sigma = [\frac{8.72}{-5.285}\frac{-5.285}{8.72}]$

P(y1, y2|z=P) = $\frac{1}{2\pi\sqrt{0.02077}}exp(-\frac{1}{2}[y1 - 0.4, y2 - 0.4][\frac{8.72}{-5.285}\frac{-5.285}{8.72}][\frac{y1-0.4}{y2-0.4}])$

}

z = N, y1y2{

$\mu = \frac{1}{4}([\frac{1}{0}] + [\frac{1}{0}] + [\frac{0}{1}] + [\frac{1}{1}]) = [\frac{0.75}{0.5}]$

var(y1) = $\frac{1}{3}\sum_{i=1}^{4}(y_{1i} - \mu)^2 = ... = 0.25$; var(y2) = … = $\frac{1}{3}$

cov(y1, y2)= $\frac{1}{3}\sum_{i=1}^{4}(y_{1i} - \mu)(y_{2i} - \mu) = ... = -\frac{1}{6}$

$\Sigma = [\frac{0.25 \quad -\frac{1}{6}}{-\frac{1}{6} \quad \frac{1}{3}}]$; $|\Sigma| = \frac{1}{18}$; $\Sigma^{-1} = 18 * \Sigma = [\frac{6}{3}\frac{3}{4.5}]$

P(y1, y2|z=N) = $\frac{1}{2\pi\sqrt{\frac{1}{18}}}exp(-\frac{1}{2}[y1 - 0.75, y2 - 0.5][\frac{6}{3}\frac{3}{4.5}][\frac{y1-0.75}{y2-0.5}])$

}

z = P, y3{

$\mu = \frac{1}{5}(1 + 0.9 + 1.2 + 0.8) = 0.84$

$\sigma^2 = \frac{1}{4}\sum_{i=1}^{5}(y_{3i} - \mu)^2 = ... \approx 0.063$

P(y3|z = P) = $\frac{1}{\sqrt{2\pi*0.063}}exp(-\frac{1}{2*0.063}(y_3 - 0.84)^2)$

}

z = N, y3{

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

$\sigma^2 = \frac{1}{3}\sum_{i=1}^{4}(y_{3i} - \mu)^2 = ... \approx 0.029$

P(y3|z = N) = $\frac{1}{\sqrt{2\pi*0.029}}exp(-\frac{1}{2*0.029}(y_3 - 0.975)^2)$

}

3) (usando as expressões obtidas na alínea anterior)

$p(c = P|x_{new1}) = \frac{p(y_1=A,y_2=1|c=P)*p(y_3=0.3|c=P)*p(c=P)}{p(y_1=A,y_2=1|c=P)*p(y_3=0.8|c=P)*p(c=P)+p(y_1=A,y_2=1|c=N)*p(y_3=0.8|c=N)*p(c=N)} \approx 0.0618$

$p(c = P|x_{new2}) = \frac{p(y_1=B,y_2=1|c=P)*p(y_3=1|c=P)*p(c=P)}{p(y_1=B,y_2=1|c=P)*p(y_3=1|c=P)*p(c=P)+p(y_1=B,y_2=1|c=N)*p(y_3=1|c=N)*p(c=N)} \approx$ 0.29104

$p(c = P|x_{new3}) = \frac{p(y_1=B,y_2=0|c=P)*p(y_3=0.9|c=P)*p(c=P)}{p(y_1=B,y_2=0|c=P)*p(y_3=0.9|c=P)*p(c=P)+p(y_1=B,y_2=0|c=N)*p(y_3=0.9|c=N)*p(c=N)}$ ≈0.00439

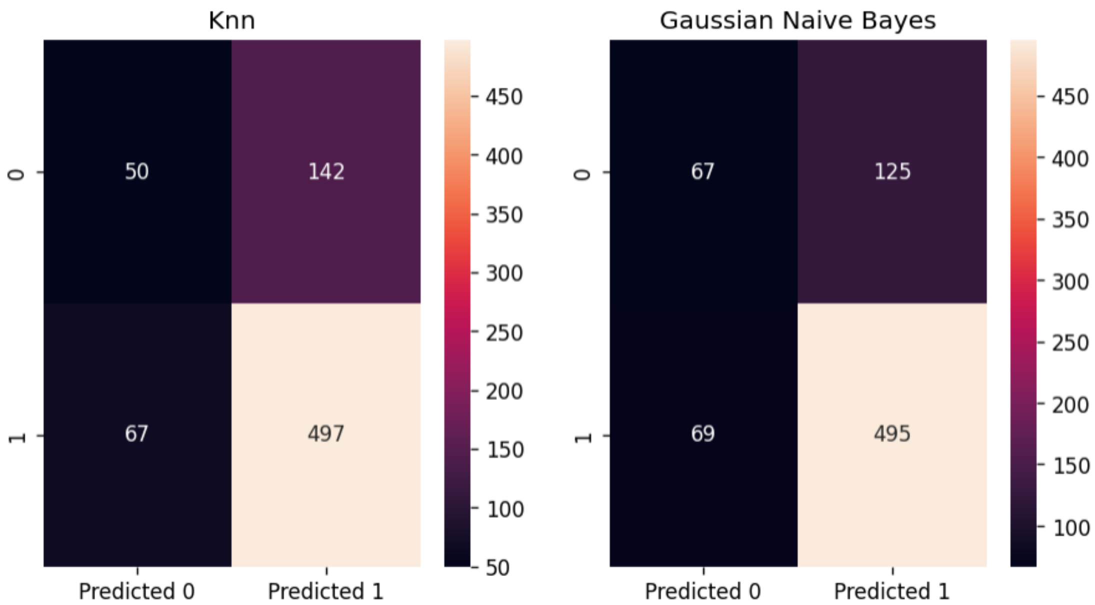
II. Programming and critical analysis [7v]

```
# Import Wall
import pandas as pd
import numpy as np
from sklearn import metrics, datasets, tree
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.model_selection import StratifiedKFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
import matplotlib.pyplot as plt
from scipy.io.arff import loadarff
import seaborn as sns
from scipy import stats

data = loadarff('pd_speech.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')

X = df.iloc[:, :-1]
Y = df.iloc[:, -1]
inputs = df.drop('class', axis=1).values
outputs = df['class'].values

# stratified k-fold cross validation
skf = StratifiedKFold(n_splits=10, shuffle=True, random_state=0)

# predict using Knn
predictorKnn = KNeighborsClassifier(n_neighbors=5, metric='euclidean',
weights='uniform')
# predict using gaussian naive bayes
predictorGnb = GaussianNB()

cm = None
m = []
accuracyKnn = []
accuracyNB = []
for train, test in skf.split(X, Y):
    X_train, X_test = X.iloc[train], X.iloc[test]
    Y_train, Y_test = Y.iloc[train], Y.iloc[test]
    predictorKnn.fit(X_train, Y_train)
```

```
    Y_pred = predictorKnn.predict(X_test)
    accuracyKnn.append(accuracy_score(Y_test, Y_pred))
    if type(cm) == type(None):
        cm = np.array(confusion_matrix(Y_test, Y_pred, labels=['0',
'1']))
    else:
        cm +=  np.array(confusion_matrix(Y_test, Y_pred, labels=['0',
'1']))

result = pd.DataFrame(cm, index=['0', '1'], columns=['Predicted 0',
'Predicted 1'])
m.append(result)


cm = None
for train, test in skf.split(X, Y):
    X_train, X_test = X.iloc[train], X.iloc[test]
    Y_train, Y_test = Y.iloc[train], Y.iloc[test]
    predictorGnb.fit(X_train, Y_train)
    Y_pred = predictorGnb.predict(X_test)
    accuracyNB.append(accuracy_score(Y_test, Y_pred))
    if type(cm) == type(None):
        cm = np.array(confusion_matrix(Y_test, Y_pred, labels=['0',
'1']))
    else:
        cm +=  np.array(confusion_matrix(Y_test, Y_pred, labels=['0',
'1']))

result = pd.DataFrame(cm, index=['0', '1'], columns=['Predicted 0',
'Predicted 1'])
m.append(result)


#plot confusion matrix
fig, ax = plt.subplots(1, 2, figsize=(10, 5))
sns.heatmap(m[0], annot=True, ax=ax[0], fmt='d')
sns.heatmap(m[1], annot=True, ax=ax[1], fmt='d')
ax[0].set_title('Knn')
ax[1].set_title('Gaussian Naive Bayes')
plt.show()

# Knn p value > NB p value
res = stats.ttest_rel(accuracyKnn, accuracyNB, alternative='greater')
```

```
print('Knn > Nb -> p-value: ', res.pvalue)
# Knn p value > NB p value
res = stats.ttest_rel(accuracyKnn, accuracyNB, alternative='less')
print('Knn < Nb -> p-value: ', res.pvalue)
```

5)



6)

Applying the T-Student formula for the hypothesis H0: "kNN is statistically superior to Naïve Bayes", and H1 being the null hypothesis, we'll get the following values:

```
Knn > Nb -> p-value:  0.9104476998751558
Knn < Nb -> p-value:  0.08955230012484414
```

In this way after seeing the values , since the p-value of the hypothesis H1 is lower than 0.10 this indicates strong evidence against the null hypothesis therefore validating the H0 statement.

7) Differences in performance between kNN and Naïve Bayes can be caused by:

- Naïve Bayes makes the assumption that all variables are independent, and there is no guarantee that this is the case in this context.
- kNN is a simple algorithm when compared to Naïve Bayes, but it is quite accurate when handling datasets of small samples, even more than Naïve Bayes, that requires a larger, more complex set of data to be more accurate.

- kNN has better performance when we normalize the data to the same scale that way the distance is more coherent with the classification.

END