TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 35

I. Pen-and-paper

1) $P(c = 0 \mid xnew) = \frac{P(xnew \mid c=0)\,P(c=0)}{P(xnew)} = \frac{P(y1 \mid c=0)\,P(y2 \mid c=0)\,P(y3,y4 \mid c=0)\,P(c=0)}{P(xnew)}$

- $P(y1 \mid c = 0)$ $\quad y1 \sim N(\mu', \sigma^{2\prime}) \Rightarrow y1 \mid c = 0 \sim N(\mu, \sigma^2)$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} y1_i = \frac{1}{4}\sum_{i=1}^{4} y1_i = \frac{0.6+0.1+0.2+0.1}{4} = 0.25$$

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y1_i - \mu)^2 = \frac{1}{3}\sum_{i=1}^{4}(y1_i - \mu)^2 = \frac{(0.6-0.25)^2 + (0.1-0.25)^2 + \dots}{3} = 0.0567$$

$\therefore y1 \mid c = 0 \sim N(\mu = 0.25, \sigma^2 = 0.0567)$

- $P(y2 \mid c = 0)$

$$P(y2 = A \mid c = 0) = \frac{1}{2}$$

$$P(y2 = B \mid c = 0) = P(y2 = C \mid c = 0) = \frac{1}{4}$$

- $P(y3, y4 \mid c = 0)$ $\quad \{y3, y4\} \sim N(\mu', \Sigma') \Rightarrow y3, y4 \mid c = 0 \sim N(\mu, \Sigma)$

$$\mu_3 = \frac{1}{4}\sum_{i=1}^{4} y3_i = \frac{0.2-0.1-0.1+0.8}{4} = 0.2$$

$$\mu_4 = \frac{0.4-0.4+0.2+0.8}{4} = 0.25$$

$$\sigma_3^2 = \frac{1}{3}\sum_{i=1}^{4}(y3_i - \mu_3)^2 = \frac{(0.2-0.2)^2 + (-0.1-0.2)^2 + \dots}{3} = 0.18$$

$$\sigma_4^2 = \frac{(0.4-0.25)^2 + (-0.4-0.25)^2 + \dots}{3} = 0.25$$

$$cov(y3, y4) = \frac{1}{n-1}\sum_{i=1}^{n}(y3_i - \mu_3)(y4_i - \mu_4) =$$

$$= \frac{(0.2-0.2)(0.4-0.25) + (-0.1-0.2)(-0.4-0.25) + \dots}{3} = 0.18$$

$\therefore y3, y4 \mid c = 0 \sim N(\mu = [0.2\ 0.25]^T, \Sigma = [[0.18, 0.18], [0.18, 0.25]])$

$P(c = 1 \mid xnew) = \frac{P(xnew \mid c=1)\,P(c=1)}{P(xnew)} = \frac{P(y1 \mid c=1)\,P(y2 \mid c=1)\,P(y3,y4 \mid c=1)\,P(c=1)}{P(xnew)}$

- $P(y1 \mid c = 1)$ $\quad y1 \sim N(\mu', \sigma^{2\prime}) \Rightarrow y1 \mid c = 1 \sim N(\mu, \sigma^2)$

$$\mu = \frac{1}{6}\sum_{i=1}^{6} y_{1i} = \frac{0.3 - 0.1 - 0.3 + 0.2 + 0.4 - 0.2}{6} = 0.05$$

$$\sigma^2 = \frac{1}{5}\sum_{i=1}^{6}(y_{1i} - \mu)^2 = \frac{(0.3-0.05)^2 + (-0.1-0.05)^2 + \dots}{5} = 0.083$$

$\therefore y1 \mid c = 1 \sim N(\mu = 0.05, \sigma^2 = 0.083)$

- $P(y2 \mid c = 1)$

$$P(y2 = A \mid c = 1) = \frac{1}{6} \qquad P(y2 = B \mid c = 1) = \frac{1}{3}$$

$$P(y2 = C \mid c = 1) = \frac{1}{2}$$

- $P(y3,\ y4 \mid c = 1) \qquad \{y3, y4\} \sim N(\mu', \Sigma') \Rightarrow y3, y4 \mid c = 1 \sim N(\mu, \Sigma)$

$$\mu_3 = \frac{1}{6}\sum_{i=1}^{6} y3_i = \frac{0.1 + 0.2 - 0.1 + 0.5 - 0.4 + 0.4}{6} = 0.1167$$

$$\mu_4 = \frac{0.3 - 0.2 + 0.2 + 0.6 - 0.7 + 0.3}{6} = 0.0833$$

$$\sigma_3^2 = \frac{1}{5}\sum_{i=1}^{5} (y3_i - \mu_3)^2 = \frac{(0.1-0.1167)^2 + (0.2-0.1167)^2 + ...}{3} = 0.1097$$

$$\sigma_4^2 = \frac{(0.3-0.0833)^2 + (-0.2-0.0833)^2 + ...}{3} = 0.2137$$

$$cov(y3,\ y4) = \frac{1}{n-1}\sum_{i=1}^{n} (y3_i - \mu_3)(y4_i - \mu_4) =$$

$$= \frac{(0.1-0.1167)(0.3-0.0833) + (0.2-0.1167)(-0.2-0.0833) + ...}{5} = 0.1223$$

$$\therefore y3,\ y4 \mid c = 1 \sim N(\mu = [0.1167\ \ 0.0833]^T,\ \Sigma = [[0.1097,\ 0.1223],\ [0.1223,\ 0.2137]])$$

2) Para calcular $P(c = i \mid xnew) = \frac{P(xnew \mid c = i)\, P(c = i)}{P(xnew)}$ para $i \in \{0, 1\}$ basta apenas comparar os numeradores dado que o denominador é igual em ambos os casos.

Temos $P(x1 \mid c = 0)\, P(c = 0) =$

$$= P(y1 = 0.6 \mid c = 0) P(y2 = A \mid c = 0) P(y3 = 0.2,\ y4 = 0.4 \mid c = 0) P(c = 0)$$

$$= f_{y1 \mid c=0}(0.6) \cdot \frac{1}{2} \cdot f_{y3, y4 \mid c=0}(0.2,\ 0.4)$$

$$= 0.13731$$

onde $f_{y1 \mid c=0}$ e $f_{y3, y4 \mid c=0}$ são, respetivamente, as f.d.p de $y1 \mid c = 0$ e $y3,\ y4 \mid c = 0$

$P(x1 \mid c = 1)\, P(c = 1) =$

$$= P(y1 = 0.6 \mid c = 1) P(y2 = A \mid c = 1) P(y3 = 0.2,\ y4 = 0.4 \mid c = 1)$$

$$= f_{y1 \mid c=1}(0.6) \cdot \frac{1}{2} \cdot f_{y3, y4 \mid c=1}(0.2,\ 0.4)$$

$$= 0.02713$$

Repetindo este procedimento para os vários *xnew* obtemos:

xnew	P(xnew | c = 0) P(c = 0)	P(xnew | c = 1) P(c = 1)	class
x1	0.13731	0.02713	0
x2	0.06325	0.26010	1
x3	0.23167	0.07350	0
x4	0.07041	0.08308	1
x5	0.19250	0.22942	1
x6	0.01898	0.24299	1
x7	0.00821	0.12073	1
x8	0.17782	0.20334	1
x9	0.05976	0.02569	0
x10	0.03033	0.32078	1

Confusion Matrix:

Actual \ Predicted	c = 0 (PN)	c = 1 (PP)
c = 0 (N)	True Negative : 2	False Negative : 1
c = 1 (P)	False Positive :2	True Positive : 5

3) $Precision = \frac{TP}{TP+FP} = \frac{5}{5+2} = \frac{5}{7}$ $\quad\quad$ $Recall = \frac{TP}{TP+FN} = \frac{5}{5+1} = \frac{5}{6}$

$$\therefore F1 = \left(\frac{Precision^{-1} + Recall^{-1}}{2}\right)^{-1} = 0.76923$$

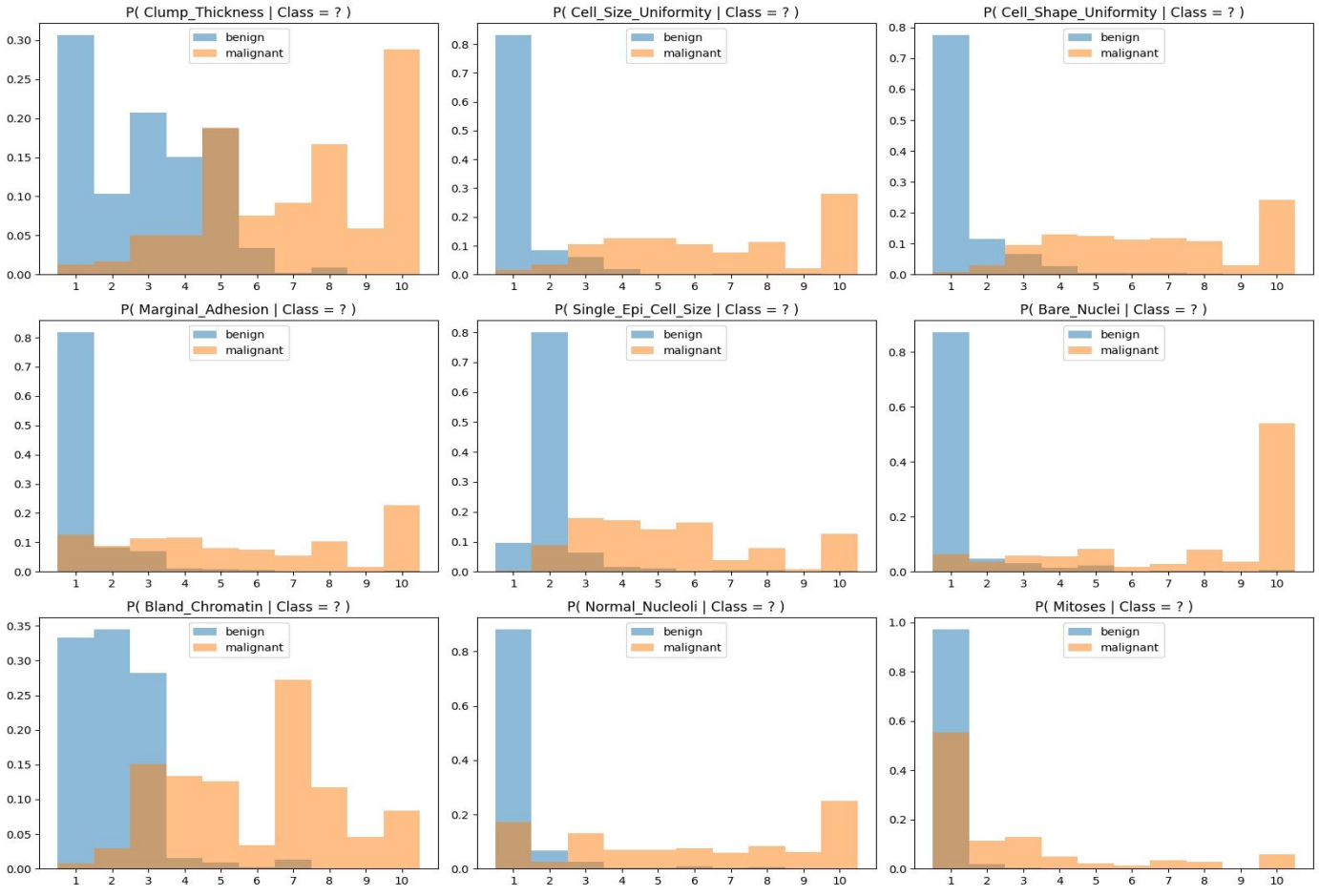
4) $P(c = 1 \mid x) = \frac{P(x \mid c=1)\,P(c=1)}{P(x)} = \frac{P(x \mid c=1)\,P(c=1)}{P(x \mid c=1)\,P(c=1) + P(x \mid c=0)\,P(c=0)}$

	Actual class	P(c = 1| x)	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
x1	0	0,16498	FP	FP	TN	TN	TN	TN	TN	TN	TN	TN	TN
x2	0	0,80493	FP	FP	FP	FP	FP	FP	FP	FP	FP	TN	TN
x3	0	0,24086	FP	FP	FP	TN	TN	TN	TN	TN	TN	TN	TN
x4	0	0,54128	FP	FP	FP	FP	FP	FP	TN	TN	TN	TN	TN
x5	1	0,54375	TP	TP	TP	TP	TP	TP	FN	FN	FN	FN	FN
x6	1	0,92755	TP	TP	TP	TP	TP	TP	TP	TP	TP	TP	FN
x7	1	0,93634	TP	TP	TP	TP	TP	TP	TP	TP	TP	TP	FN
x8	1	0,53349	TP	TP	TP	TP	TP	TP	FN	FN	FN	FN	FN
x9	1	0,30066	TP	TP	TP	TP	FN	FN	FN	FN	FN	FN	FN
x10	1	0,91362	TP	TP	TP	TP	TP	TP	TP	TP	TP	TP	FN
	Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	60%	60%	70%	**80%**	70%	70%	60%	60%	60%	70%	40%

Com o threshold a 0.3, a *accuracy* do classificador é 80%, o que significa que com o mesmo, o classificador consegue prever corretamente 80% dos casos. Para valores abaixo do threshold, estes são classificados como classe 1 e, para valores superiores, como classe 0.

II. Programming and critical analysis

5) (Código 1)



6) O problema de overfitting ocorre quando um classificador, apesar de ter uma boa performance com os seus dados de treino, obtém uma má performance com novas observações.

Através do nosso programa em Python (Código 2), obtivemos as seguintes *accuracies* médias de *kNN*: 0.970737 para k=3, 0.970716 para k=5 e 0.969288 para k=7. Os resultados obtidos mostram que o k menos suscetível a overfitting é k=3, uma vez que é o que apresenta uma ligeira maior *accuracy* após realizar a *cross-validation*.

É de notar que, geralmente, k=3 apresenta um risco de overfitting relativamente elevado para uma amostra deste tamanho. No nosso caso, devido à seed usada na função KFold, os resultados para k=3 são mais exatos em comparação com k=5 e k=7.

7) (Código 2) Fixando k=3 no modelo *kNN* e mantendo todas as condições presentes no enunciado anterior, calculámos a *accuracy* dos resultados obtidos. O mesmo fizemos para o modelo *Multinomial Naïve Bayes* obtendo, de igual modo, a sua *accuracy*.

Decidimos testar a Hipótese Nula (H0) "kNN é estatisticamente igual (=) a MultinomialNB" vs a Hipótese Alternativa (H1) "kNN é estatisticamente superior (>) a MultinomialNB". Para tal, utilizamos o método *ttest_rel* com ambos os vetores de *accuracy* obtidos. Através desta função, obtivemos uma *t-statistic* de 3.80082 e um *p-value* de 0.00955. Deste modo, como o *p-value* é bastante baixo e inferior aos níveis de significância usuais (0.05 e 0.10), rejeita-se H0, ou seja, confirma-se H1: "kNN é estatisticamente superior a MultinomialNB".

8) Tendo por base os resultados obtidos, conseguimos concluir que duas razões que estão na origem de uma diferença de performance entre *kNN* e *Naïve Bayes* são o facto do modelo *Naïve Bayes* assumir independência entre todos os atributos/características e o facto do modelo *kNN* ser inerentemente mais otimizado para decidir localmente e, por isso, melhor para encontrar semelhanças entre observações.

III. APPENDIX

Código 1

```
import numpy as np, pandas as pd, matplotlib.pyplot as plt
from scipy.io import arff

data = arff.loadarff('breast.w.arff')
df = pd.DataFrame(data[0])
df = df.dropna()

#separating benigns and malignant lists
grouped = df.groupby(['Class'])
benign = grouped.get_group(b'benign')
malignant = grouped.get_group(b'malignant')

test = pd.DataFrame(np.random.randn(30,9), columns=map(str, range(9)))
fig = plt.figure(figsize=(16,12))

for i in range(9):
    plt.subplot(3,3,i+1)
    bins = np.linspace(1, 11, num = 11)
    weights_benign = np.ones_like(benign.iloc[:,i]) / (len(benign.iloc[:,i]))
    plt.hist(benign.iloc[:,i].values, bins, align="left", alpha = 0.5, label = 'benign',
                                                                    weights=weights_benign)
    weights_malignant = np.ones_like(malignant.iloc[:,i]) / (len(malignant.iloc[:,i]))
    plt.hist(malignant.iloc[:,i].values, bins, align="left", alpha = 0.5, label = 'malignant',
                                                                    weights=weights_malignant)
    plt.xticks(range(1,11))
    plt.legend(loc = 'upper center')
    plt.title("P( {} | Class = ? )".format(df.columns[i]))

fig.tight_layout()  # Improves appearance a bit.
fig.savefig("graphs.jpg")
```

Código 2

```
import pandas as pd
from statistics import mean
from scipy.io import arff
from scipy.stats import ttest_rel
from sklearn.model_selection import KFold, cross_val_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier

# Conditions
FOLDS = 10
GROUP_NUMBER = 35
k = [3,5,7]
EUCLIDEAN_DIST= 2
WEIGHTS = "uniform"

data = arff.loadarff('breast.w.arff')
df = pd.DataFrame(data[0])
df = df.dropna()
data = df.drop(columns=["Class"]).values
results = df[df.keys()[-1]].astype('string').values

cross_val = KFold(n_splits=FOLDS, shuffle=True, random_state=GROUP_NUMBER)

print("#######  EX. 6 #######\n")
for i in k:
    model = KNeighborsClassifier(n_neighbors=i, weights=WEIGHTS, p=EUCLIDEAN_DIST)
    cv_scores = cross_val_score(model, data, results, scoring='accuracy', cv=cross_val)
    print(" - Accuracies (k={}): {} \n\t ## Mean Accuracy: {}\n".format(i,cv_scores,mean(cv_scores)))

print("\n#######  EX. 7 #######\n")
modelKNN = KNeighborsClassifier(n_neighbors=3, weights=WEIGHTS, p=EUCLIDEAN_DIST)
accuracyKNN = cross_val_score(modelKNN, data, results)
modelNB = MultinomialNB()
accuracyNB = cross_val_score(modelNB, data, results)
print("statistic = {} ; p-value = {}".format(ttest_rel(accuracyKNN, accuracyNB,
               alternative="greater")[0], ttest_rel(accuracyKNN, accuracyNB, alternative="greater")[1]))
```

Programa auxiliar (usado na parte I do trabalho):
https://github.com/martimfasantos/Apre/blob/main/homework1/bayes.py

END