TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 024

I. Pen-and-paper

1) Para treinar um classificador bayesiano, é necessário calcular:

$$\frac{P(N|x_{new})}{P(P|x_{new})}$$

Ou seja, comparar a probabilidade de ocorrer *N* e a de ocorrer *P*, sabendo que ocorreu x_{new}. Se o valor referido for superior a 1, x_{new} é classificado como *N*, caso contrário, como *P*. Considerando os valores dados, para $x_{new} = x_1$:

$$P(N|x_1) = \frac{P(x_1|N) \cdot P(N)}{P(x_1)} \quad (Teorema\ de\ Bayes)$$

Onde $P(N) = \frac{4}{10} = 0.4$, $P(x_1)$ não influencia o classificador, e:

$$P(x_1|N) = P(y_1 = 0.6|N) \cdot P(y_2 = A|N) \cdot P\big((y_3 = 0.2, y_4 = 0.4)\big|N\big)$$

Pois existe independência entre $\{y_1\}, \{y_2\}$ e $\{y_3, y_4\}$. Para y_1, pela definição de distribuição normal:

$$P(y_1 = 0.6|N) = \frac{1}{\sqrt{2 \cdot \pi} \cdot \sigma} \cdot exp\left(-\frac{1}{2 \cdot \sigma^2} \cdot (y_1 - \mu)^2\right)$$

Sendo que μ e σ^2 são, respetivamente, estimados pela média e pela variância corrigida (S^2):

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_{1i} - \bar{y}_1)^2$$

Obtendo-se o valor:

$$P(y_1 = 0.6|N) \approx 0.5686237446173313$$

Dado que y_2 é uma variável categórica, a probabilidade associada é calculada pelo seu número de ocorrências, a dividir pelo total de ocorrências:

$$P(y_2 = A|N) = \frac{2}{4} = 0.5$$

Calculando a última expressão através da definição, sendo $D = 2$ (dimensão da matriz):

$$P\big((y_3 = 0.2, y_4 = 0.4)\big|N\big) = \frac{1}{(2 \cdot \pi)^{D/2}} \cdot \frac{1}{|\Sigma|^{1/2}} \cdot \exp\left(-\frac{1}{2} \cdot (x - \mu)^T \Sigma^{-1} \cdot (x - \mu)\right)$$

Em que:

$$x = \begin{bmatrix} y_3 = 0.2 \\ y_4 = 0.4 \end{bmatrix}$$

$$\mu = \begin{bmatrix} \bar{y}_3 \\ \bar{y}_4 \end{bmatrix}$$

$$\Sigma = \begin{pmatrix} \Sigma_{33} & \Sigma_{43} \\ \Sigma_{34} & \Sigma_{44} \end{pmatrix} \quad (matriz\ de\ covariâncias)$$

Σ_{33} equivale à variância de y_3, e Σ_{44} à de y_4. $\Sigma_{43} = \Sigma_{34}$ representam a covariância entre y_3 e y_4, podendo ser calculadas por:

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 024

$$\Sigma_{ij} = \frac{1}{n-1}\sum_{k=1}^{n}(y_{ik} - \bar{y}_i)(y_{jk} - \bar{y}_j)$$

$|\Sigma|$ denota o determinante de Σ, e Σ^{-1} a matriz inversa. Obtém-se assim:

$$P\big((y_3 = 0.2, y_4 = 0.4)\big|N\big) \approx 1.2073620797939844$$

Sendo o resultado final (excluindo $P(x_1)$):

$$P(N|x_1) \approx 0.13730694938428492$$

Pelo mesmo raciocínio, obteve-se os seguintes valores:

x_{new}	$P(N\|x_{new})$	$P(P\|x_{new})$	Classificação
x_1	0.1373	0.0272	N
x_2	0.0633	0.2607	P
x_3	0.2317	0.0737	N
x_4	0.0704	0.0831	P
x_5	0.1925	0.2292	P
x_6	0.0190	0.2430	P
x_7	0.0082	0.1207	P
x_8	0.1778	0.2031	P
x_9	0.0598	0.0257	N
x_{10}	0.0303	0.3208	P

2) Sabe-se que de x_1 até x_4 se tratam de variáveis classificadas como *N*, e as restantes como *P*, pelo que este é o seu valor real. A seguinte tabela (de confusão) estabelece a relação entre os valores reais e os previstos na alínea anterior:

Real\Previsto	N	P
N	TN = 2	FP = 2
P	FN = 1	TP = 5

Onde *TN – True N*; *TP – True P*; *FN – False N*; *FP – False P*.

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 024

3) Avaliando o *score* de F_1 para P:

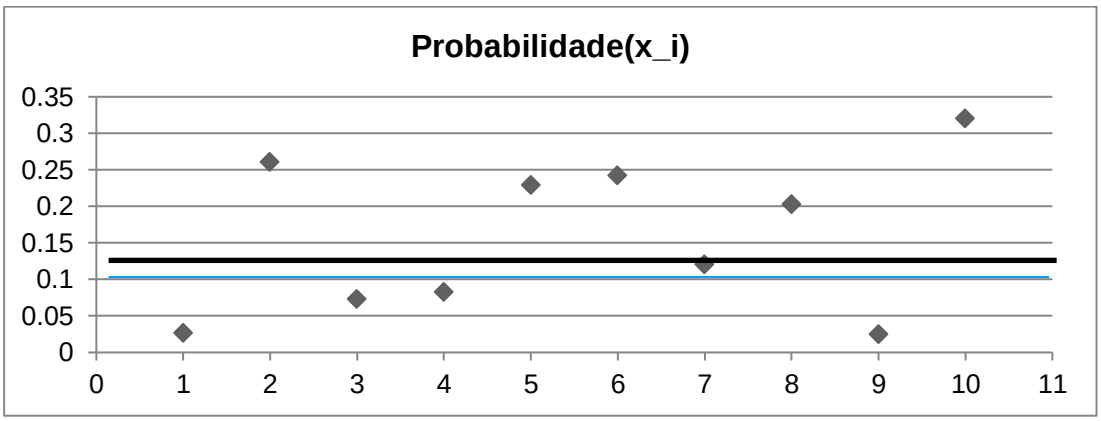
$$F_{1P} = \frac{2}{\frac{1}{sensibilidade_P} + \frac{1}{precisão_P}}$$

$$sensibilidade_P = \frac{TP}{TP + FN}$$

$$precisão_P = \frac{TP}{TP + FP}$$

$$F_{1P} \approx 0.76923$$

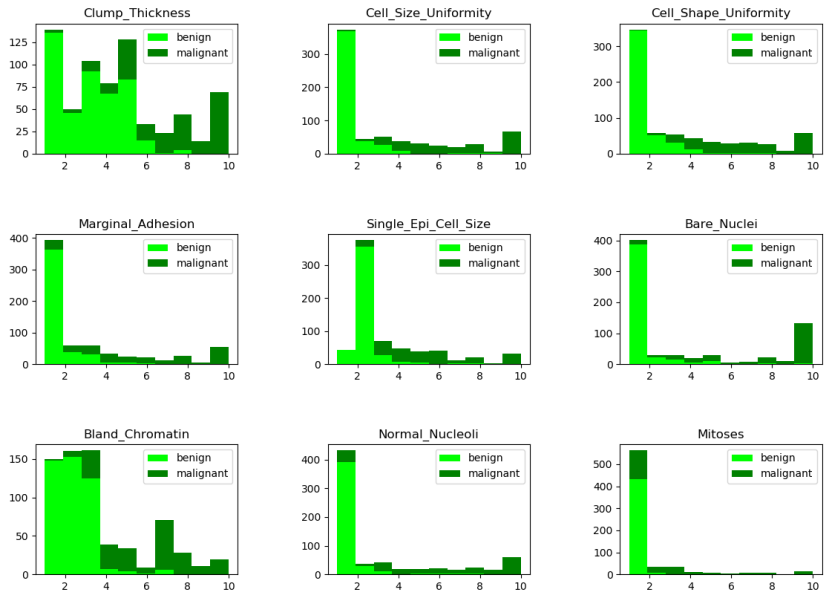
4) O seguinte gráfico representa a probabilidade calculada para cada x_i:



As linhas a azul e a negrito representam, respetivamente, as *thresholds* obtida e ideal. A ideal situa-se no intervalo [0.0831; 0.1207[uma vez que, em relação à obtida, passa a classificar x_4 como N (que é a sua classificação real). Caso estivesse mais acima, passaria a classificar x_7 como N, o que não corresponde à classificação real. Utilizando classificadores bayesianos não é possível classificar correctamente x_2 nem x_9 (*outliers*).

II. Programming and critical analysis

5)



TÉCNICO LISBOA

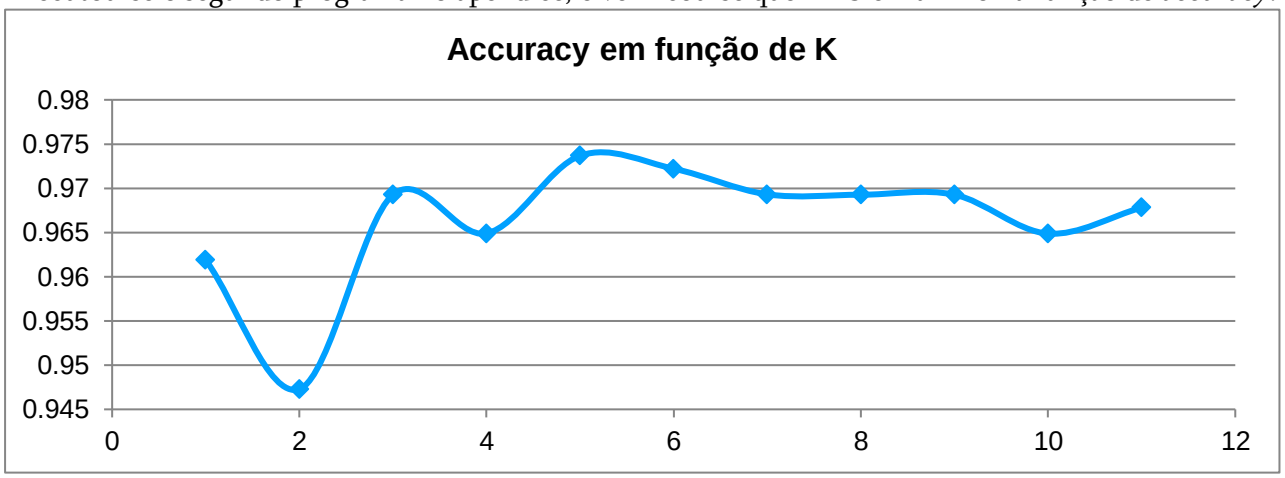
Aprendizagem 2021/22

Homework I – Group 024

6)

k	3	5	7
Accuracy	0.9692668371696505	0.9736786018755328	0.9693094629156012

Executou-se o segundo programa no apêndice, e verificou-se que k = 5 é máximo na função de *accuracy*:



Após estes valores, a função é monótona decrescente (em sentido lato). Conclui-se que k = 5 é o valor menos suscetível ao *overfitting risk*.

7) Para testar a hipótese: *"kNN is statistically superior to Naïve Bayes (multinomial assumption)"*, executou-se o terceiro programa no apêndice, e obteve-se um valor-p = $1.9691654 \cdot 10^{-5}$. Assim, rejeita-se a hipótese inicial para níveis de significância superiores ao valor-p, e não se rejeita caso contrário. Para níveis usuais de significância, *1, 5 e 10%*, rejeita-se a hipótese.

8) Com base nos resultados anteriores, verifica-se que o *Naïve Bayes* não consegue utilizar a amostra de teste a *10%*, uma vez que esta é demasiado pequena quando comparada com aqueles que deveria necessitar (cerca de *90%*). Assim, o *kNN* estima melhor os resultados para a amostra.

Outro aspeto é a assunção de independência entre variáveis por parte do *Naïve Bayes* o que, por observação dos resultados, não se deve traduzir na realidade.

III. APPENDIX

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

cancer = arff.loadarff(r'breast.w.arff')
df = pd.DataFrame(cancer[0])
df.dropna(inplace=True)
class1 = df['Class'][0]
dfc1 = df.loc[df['Class'] == class1]
```

```
dfc2 = df.loc[df['Class'] != class1]
n_bins = 10

x1 = [dfc1['Clump_Thickness'], dfc2['Clump_Thickness']]
x2 = [dfc1['Cell_Size_Uniformity'], dfc2['Cell_Size_Uniformity']]
x3 = [dfc1['Cell_Shape_Uniformity'], dfc2['Cell_Shape_Uniformity']]
x4 = [dfc1['Marginal_Adhesion'], dfc2['Marginal_Adhesion']]
x5 = [dfc1['Single_Epi_Cell_Size'], dfc2['Single_Epi_Cell_Size']]
x6 = [dfc1['Bare_Nuclei'], dfc2['Bare_Nuclei']]
x7 = [dfc1['Bland_Chromatin'], dfc2['Bland_Chromatin']]
x8 = [dfc1['Normal_Nucleoli'], dfc2['Normal_Nucleoli']]
x9 = [dfc1['Mitoses'], dfc2['Mitoses']]

fig, ((ax1, ax2, ax3), (ax4, ax5, ax6), (ax7, ax8, ax9)) = plt.subplots(nrows=3, ncols=3)
classes = ['benign','malignant']
colors = ['lime', 'green']
ax1.hist(x1, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax1.legend()
ax1.set_title('Clump_Thickness')

ax2.hist(x2, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax2.legend()
ax2.set_title('Cell_Size_Uniformity')

ax3.hist(x3, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax3.legend()
ax3.set_title('Cell_Shape_Uniformity')

ax4.hist(x4, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax4.legend()
ax4.set_title('Marginal_Adhesion')

ax5.hist(x5, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax5.legend()
ax5.set_title('Single_Epi_Cell_Size')

ax6.hist(x6, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax6.legend()
ax6.set_title('Bare_Nuclei')

ax7.hist(x7, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax7.legend()
ax7.set_title('Bland_Chromatin')

ax8.hist(x8, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax8.legend()
ax8.set_title('Normal_Nucleoli')
```

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 024

```
ax9.hist(x9, n_bins, histtype='bar', color=colors, stacked=True, label=classes)
ax9.legend()
ax9.set_title('Mitoses')

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

```
from scipy.io import arff
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold

cancer = arff.loadarff(r'breast.w.arff')
df = pd.DataFrame(cancer[0])
df.dropna(inplace=True)

i = 0
df = df.replace(df['Class'][0],0)
while df['Class'][i] == 0:
    i += 1
df = df.replace(df['Class'][i],1)

x=df[['Clump_Thickness','Cell_Size_Uniformity','Cell_Shape_Uniformity','Marginal_Adhesion','Single_E
pi_Cell_Size','Bare_Nuclei','Bland_Chromatin','Normal_Nucleoli','Mitoses']]
y = df['Class']
knn_cv = KFold(n_splits=10, random_state=24, shuffle=True)

for k in range(1, 200):
    knn = KNeighborsClassifier(n_neighbors=k, metric='euclidean', weights='uniform')
    scores = cross_val_score(knn, x, y, cv=knn_cv, scoring='accuracy')
    print(scores)
    print(k, "mean=",scores.mean(), "\n")
```

```
from scipy.io import arff
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from scipy.stats import ttest_ind
from sklearn.naive_bayes import MultinomialNB

cancer = arff.loadarff(r'breast.w.arff')
df = pd.DataFrame(cancer[0])
df.dropna(inplace=True)
```

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 024

```
i = 0
df = df.replace(df['Class'][0],0)
while df['Class'][i] == 0:
    i += 1
df = df.replace(df['Class'][i],1)

xdf[['Clump_Thickness','Cell_Size_Uniformity','Cell_Shape_Uniformity','Marginal_Adhesion','Single_Epi_Cell_Size','Bare_Nuclei','Bland_Chromatin','Normal_Nucleoli','Mitoses']]
y = df['Class']
knn_cv = KFold(n_splits=10,random_state=24,shuffle=True)

k=3
knn = KNeighborsClassifier(n_neighbors=k, metric='euclidean', weights='uniform')
scores = cross_val_score(knn, x, y, cv=knn_cv, scoring='accuracy')
print(scores)
print(k, "mean=",scores.mean(), "\n")

nb = MultinomialNB()

nb_scores = cross_val_score(nb, x, y, cv=knn_cv, scoring='accuracy')
print(nb_scores)
ttest,pval=ttest_ind(scores,nb_scores)
print("pval",pval)
```

END