TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 010

I. Pen-and-paper

1) $P(class = 0 \mid x_{new}) = \frac{P(x_{new} \mid class = 0) \times P(class = 0)}{P(x_{new})} = \frac{P(y1 \mid c = 0) \times P(y2 \mid c = 0) \times P(y3, y4 \mid c = 0) \times P(class = 0)}{P(x_{new})}$

- $P(y1 \mid c = 0)$: $\quad y1 \mid c = 0 \sim N(\mu, \delta^2)$

  $\mu = \frac{\sum_1^n y_{1i}}{n} = \frac{0.6 + 0.1 + 0.2 + 0.1}{4} = 0.25 \quad \delta^2 = \frac{\sum_1^4 (y_{1i} - \mu)^2}{n-1} = \frac{(0.6 - 0.25)^2 + \dots}{3} = 0.0567$

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

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

  $P(y2 = A \mid c = 0) = \frac{2}{4} = 0.5 \quad P(y2 = B \mid c = 0) = P(y2 = C \mid c = 0) = \frac{1}{4} = 0.25$

- $P(y3, y4 \mid c = 0)$: $\quad y3, y4 \mid c = 0 \sim N(\mu, \Sigma)$

  $\mu_3 = \frac{\sum_1^n y_{3i}}{n} = \frac{0.2 - 0.1 - 0.1 + 0.8}{4} = 0.2 \quad \mu_4 = \frac{\sum_1^n y_{4i}}{n} = \frac{0.4 - 0.4 + 0.2 + 0.8}{4} = 0.25$

  $\delta_3^2 = \frac{\sum_1^4 (y_{3i} - \mu_3)^2}{n-1} = \frac{(0.2-0.2)^2 + (-0.1-0.2)^2 + \dots}{3} = 0.18 \quad \delta_4^2 = \frac{\sum_1^4 (y_{4i} - \mu_4)^2}{n-1} = \frac{(0.4-0.25)^2 + (-0.4-0.25)^2 + \dots}{3} = 0.25$

  $cov(y3, y4) = \frac{\sum_1^4 (y_{3i} - \mu_3)(y_{4i} - \mu_4)}{n-1} = \frac{(0.2-0.2) \times (0.4-0.25) + (-0.1-0.2) \times (-0.4-0.25) + \dots}{3} = 0.18$

  $y3, y4 \mid c = 0 \sim N(\mu = \begin{bmatrix} 0.2 \\ 0.25 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.18 & 0.18 \\ 0.18 & 0.25 \end{bmatrix})$

- $P(c = 0) = \frac{4}{10} = 0.4$

$P(class = 1 \mid x_{new}) = \frac{P(x_{new} \mid class = 1) \times P(class = 1)}{P(x_{new})} = \frac{P(y1 \mid c = 1) \times P(y2 \mid c = 1) \times P(y3, y4 \mid c = 1) \times P(class = 1)}{P(x_{new})}$

- $P(y1 \mid c = 1)$: $\quad y1 \mid c = 1 \sim N(\mu, \delta^2)$

  $\mu = \frac{\sum_1^n y_{1i}}{n} = \frac{0.3 - 0.1 - 0.3 + 0.2 + 0.4 - 0.2}{6} = 0.05 \quad \delta^2 = \frac{\sum_1^6 (y_{1i} - \mu)^2}{n-1} = \frac{(0.3-0.05)^2 + \dots}{5} = 0.083$

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

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

  $P(y2 = A \mid c = 1) = \frac{1}{6} \quad P(y2 = B \mid c = 1) = \frac{2}{6} = \frac{1}{3} \quad P(y2 = C \mid c = 1) = \frac{3}{6} = 0.5$

Aprendizagem 2021/22

Homework I – Group 010

- $P(y3,\ y4 \mid c\ =\ 0)$: $\quad y3,\ y4 \mid c\ =\ 0 \sim\ N(\mu, \Sigma)$

$$\mu_3 = \frac{\sum_1^n y_{3i}}{n} = \frac{0.1+0.2-0.1+0.5-0.4+0.4}{6} = 0.1167 \quad \mu_4 = \frac{\sum_1^n y_{4i}}{n} = \frac{0.3-0.2+0.2+0.6-0.7+0.3}{6} = 0.083$$

$$\delta_3^2 = \frac{\sum_1^4 (y_{3i} - \mu_3)^2}{n-1} = \frac{(0.1-0.1167)^2+(0.2-0.1167)^2+\dots}{5} = 0.11$$

$$\delta_4^2 = \frac{\sum_1^4 (y_{4i} - \mu_4)^2}{n-1} = \frac{(0.3-0.083)^2+(-0.2-0.083)^2+\dots}{5} = 0.214$$

$$cov(y3,\ y4) = \frac{\sum_1^6 (y_{3i} - \mu_3)(y_{4i} - \mu_4)}{n-1} = \frac{(0.1-0.1167)\times(0.3-0.083)+(0.2-0.1167)\times(-0.2-0.083)+\dots}{5} = 0.122$$

$$y3,\ y4 \mid c\ =\ 0 \sim\ N\left(\mu = \begin{bmatrix} 0.1167 \\ 0.083 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.10967 & 0.122 \\ 0.122 & 0.21367 \end{bmatrix}\right)$$

- $P(c\ =\ 1)\ =\ \frac{6}{10} = 0.6$

2) Legenda: $P0_{xnew} = P(class\ =\ 0 \mid x_{new}) \times P(x_{new}) \quad P1_{xnew} = P(class\ =\ 1 \mid x_{new}) \times P(x_{new})$

	$P0_{xnew}$	$P1_{xnew}$
x_1	**0.137**	0.027
x_2	0.063	**0.261**
x_3	**0.232**	0.074
x_4	0.070	**0.083**
x_5	0.193	**0.229**
x_6	0.019	**0.243**
x_7	0.008	**0.121**
x_8	0.178	**0.203**
x_9	**0.060**	0.026
x_{10}	0.030	**0.321**

Predicted \ Actual	Class = 0	Class = 1
Class = 0	2	1
Class = 1	2	5

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 010

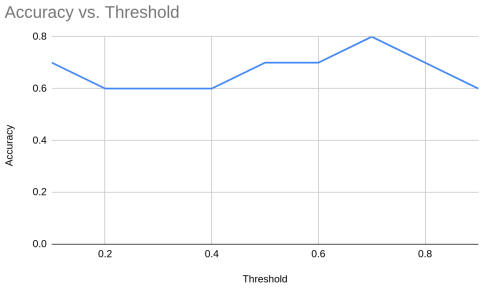
3) $P = Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} = \frac{5}{5+2} = \frac{5}{7}$

$R = Recall = \frac{True\ Positives}{Positives} = \frac{5}{6}$ $\frac{1}{F} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R}) = \frac{1}{2}(\frac{7}{5} + \frac{6}{5}) = \frac{13}{10} = 1.3$

$F = \frac{1}{1.3} = 0.769$

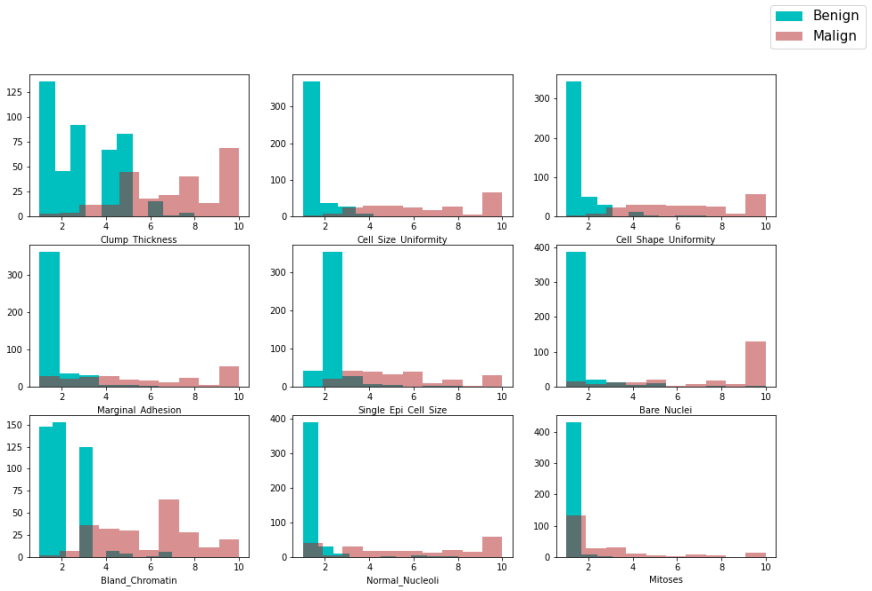
4) The default threshold usually used is 0.5. However, sometimes there is a better threshold value that makes predictions more accurate. Instead of just comparing $P(x_{new} \mid class = 0) \times P(class = 0)$ and $P(x_{new} \mid class = 1) \times P(class = 1)$ to see which one is the highest, we adopt a new method : if $\frac{P(x_{new} \mid class=0)}{P(x_{new} \mid class=0) + P(x_{new} \mid class=1)}$ >= threshold it means that x_{new} is classified as $class = 0$. Otherwise it is classified as $class = 1$. After drawing an 'accuracy vs threshold' graphic we can conclude that the best threshold value is 0.7 because it is the one with the highest accuracy and best values for the true positive rate (TPR) and false positive rate (FPR) together, making the most accurate prediction.

Threshold	FPR	TPR
0.9	1	1
0.8	0.75	1
0.7	0.5	1
0.6	0.5	0.833
0.5	0.5	0.833
0.4	0.25	0.5
0.3	0.25	0.5
0.2	0.25	0.5
0.1	0	0.5



II. Programming and critical analysis

5)



TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 010

6)

	Accuracy
K = 3	0.9706948
K = 5	0.9736147
K = 7	0.9736573

When deciding the best value for K we need to be careful to choose one large enough to avoid overfitting, but small enough to avoid oversimplifying the distribution. With the accuracy values obtained by applying KNN to the dataset we can conclude that K = 7 is the best value and as it is the one that best models the population, it is the least susceptible to the overfitting risk.

7) After using a 10-fold cross validation to split the data for training and testing the KNN and the Naïve Bayes classifiers on the same subsets and applying the t-test we can reject the hypothesis that "*k*NN is statistically equal to Naïve Bayes", and by that we can confirm the alternative hypothesis "*k*NN is statistically superior to Naïve Bayes (multinomial assumption)". The p-value obtained is greater or equal than 0.5 and because of that it is statistically significant and indicates strong evidence for the alternative hypothesis.

8) By analyzing the results obtained on the previous questions we can conclude that for this dataset the KNN classifier (accuracy = 97.068%) has a better performance than the Naïve Bayes (accuracy = 96.189%).

Each classifier has its own advantages and disadvantages and those depend a lot on the criteria used as input as well as the constraint and implementation.

The Naïve Bayes is optimal when the naïve assumptions (the priors) are accurate, which means that performance is sensitive to skewed data. This implies that the training data must be representative of the class distributions on the overall population, otherwise the prior estimates will not be correct. In this exercise we do not know the priors estimates *à priori* and they are estimated based on just that sample of the population that may not be representative. As we can verify, there are 239 instances from one class and 444 from the other and this difference may be causing a little decrease in the accuracy.
Furthermore, the KNN classifier does not assume independence between variables and when there is a large number of variables (which is the case), Naïve Bayes can cause the calculations to become inaccurate. Also, KNN does not require any specification of information about class probabilities since it just uses the formula for probability. This gets around the problem of badly estimated priors that influences Naïve Bayes accuracy.

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 010

III. APPENDIX

```
# -------------------------- Parse of input data file -----------------------------
from scipy import stats
import numpy as np
from sklearn.model_selection import KFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
import pandas as pd
from matplotlib import pyplot as plt
from scipy.io.arff import loadarff

raw_data = loadarff('breast.w.arff')
df_data = pd.DataFrame(raw_data[0])  # converting data to a pandas DataFrame
df_data = df_data.dropna()  # all rows with Na values are dropped
df_data['Class'].replace({b'malignant': 1, b'benign': 0}, inplace=True)

# --------------------------- Drawing the plots -------------------------------------
fig = plt.figure(figsize=(15, 10))

for idx, variable in enumerate(df_data.drop(columns='Class')):
   sub = fig.add_subplot(3, 3, idx+1) # A 3x3 plot grid is created and subplots are placed
   sub.set_xlabel(variable)
   # data is separated by class and plots are overlaid
   sub.hist(df_data[variable].loc[df_data['Class'] == 0], color="c")
   sub.hist(df_data[variable].loc[df_data['Class'] == 1], color="firebrick", alpha=0.5)

fig.legend(labels=["Benign", "Malign"], loc="upper right", fontsize=15)
plt.savefig('plots.png')

# ----------------- KNN cross validation : Finding the best K value ------------------
data = df_data.drop(columns=['Class']).values # all columns except for the class column
target = df_data['Class'].values  # target column is the class column

for k in range(3, 8, 2):  # data is split with a 10-fold cv and used for training and testing
   knn = KNeighborsClassifier(n_neighbors=k, weights="uniform", p=2)
   kf = KFold(n_splits=10, shuffle=True, random_state=10) # random_state = seed = 10
   accuracies = []
   for train_subset, test_subset in kf.split(data):
       X_train, X_test = data[train_subset], data[test_subset]
       Y_train, Y_test = target[train_subset], target[test_subset]
       knn.fit(X_train, Y_train)  # train
       accuracies.append(knn.score(X_test, Y_test))  # test and store accuracy
   print("Accuracy with K = " + str(k) + " " + str(np.mean(accuracies)))  # accuracy mean

# ----------------------------------Hypothesis Test-----------------------------------
# classifiers
knn = KNeighborsClassifier(n_neighbors=3, weights="uniform", p=2)
naive_bayes = MultinomialNB()
knn_acc, bayes_acc = [], []  # accuracies for each set and for each classifier

for train_subset, test_subset in kf.split(data):
   X_train, X_test = data[train_subset], data[test_subset]
   Y_train, Y_test = target[train_subset], target[test_subset]
   knn.fit(X_train, Y_train)  # train
   naive_bayes.fit(X_train, Y_train)
   knn_acc.append(knn.score(X_test, Y_test))  # test and store accuracy
   bayes_acc.append(naive_bayes.score(X_test, Y_test))

t_value, pvalue = stats.ttest_rel(knn_acc, bayes_acc)  # t-test
if pvalue <= 0.05:
   print('The alternative hypotesis : "kNN is statistically superior to Naïve Bayes" is
confirmed')
else:
   print('The null hypotesis : "kNN is statistically equal to Naïve Bayes" is confirmed')
```

END