Aprendizagem 2021/22

Homework I – Group 053

I. Pen-and-paper

1)

	Class = 0	Class = 1
Prior	$P(Class\ =\ 0)\ =\ \frac{4}{10} = 0.4$	$P(Class\ =\ 1)\ =\ \frac{6}{10} = 0.6$
Y1	$P(Y1 \mid Class = 0) = N(\mu_0, {\sigma^2}_0)$ $\mu_0\ = \frac{1}{4} \times \sum_{i=1}^{4} x_{i1} = 0.25$ $\sigma_0\ = \sqrt{\frac{1}{4-1} \times \sum_{i=1}^{4} (x_{i1} - \mu_0)^2} = 0.238$	$P(Y1 \mid Class = 1) = N(\mu_1, {\sigma^2}_1)$ $\mu_1\ = \frac{1}{6} \times \sum_{i=5}^{10} x_{i1} = 0.05$ $\sigma_1\ = \sqrt{\frac{1}{4-1} \times \sum_{i=5}^{10} (x_{i1} - \mu_1)^2} = 0.288$
Y2	$P(Y2 = A \mid Class = 0)\ = \frac{2}{4} = 0.5$ $P(Y2 = B \mid Class = 0)\ = \frac{1}{4} = 0.25$ $P(Y2 = C \mid Class = 0)\ = \frac{1}{4} = 0.25$	$P(Y2 = A \mid Class = 1)\ = \frac{1}{6} = 0.167$ $P(Y2 = B \mid Class = 1)\ = \frac{2}{6} = 0.333$ $P(Y2 = C \mid Class = 1)\ = \frac{3}{6} = 0.5$
Y3, Y4	$\mu_{3,4}\ = \frac{1}{4} \times \sum_{i=1}^{4} [x_{i3}\ \ x_{i4}]^T = [0.2\ \ 0.25]^T$ $\Sigma\ = \begin{pmatrix} cov(x_{i3}, x_{i3}) & cov(x_{i3}, x_{i4}) \\ cov(x_{i4}, x_{i3}) & cov(x_{i4}, x_{i4}) \end{pmatrix}$ $= \begin{pmatrix} 0.18 & 0.18 \\ 0.18 & 0.25 \end{pmatrix}$ $\beta\ = \Sigma^{-1} = \begin{pmatrix} 19.8413 & -14.2857 \\ -14.2857 & 14.2857 \end{pmatrix}$ $\lvert\Sigma\rvert\ = 0.0126$	$\mu_{3,4}\ = \frac{1}{4} \times \sum_{i=5}^{10} [x_{i3}\ \ x_{i4}]^T = [0.1167\ \ 0.0833]^T$ $\Sigma\ = \begin{pmatrix} cov(x_{i3}, x_{i3}) & cov(x_{i3}, x_{i4}) \\ cov(x_{i4}, x_{i3}) & cov(x_{i4}, x_{i4}) \end{pmatrix}$ $= \begin{pmatrix} 0.1097 & 0.1223 \\ 0.1223 & 0.2137 \end{pmatrix}$ $\beta\ = \Sigma^{-1} = \begin{pmatrix} 25.2362 & -14.4488 \\ -14.4488 & 12.9528 \end{pmatrix}$ $\lvert\Sigma\rvert\ = 0.0085$

NOTE: x_{i1} represents the feature y_1 from observation x_i.

2) For each vector of variables y in the training data we calculate the expected value using the formula $argmax\ p(c \mid y1, y2, y3, y4) = argmax\ \frac{p(y1,y2,y3,y4 \mid c)p(c)}{p(y1,y2,y3,y4)} = argmax\ p(c)\ p(y1|c)p(y2|c)p(y3, y4|c)$, assuming independence between {y1}, {y2}, and {y3,y4} and ignoring the denominator because it is independent of which class conditional distributions we are considering.

$argmax\ p(c \mid x1) = p(c = 0\ |\{y1, y2, y3, y4\} = \{0.6, A, 0.2,0.4\}) = 0.1373$ → true negative

$argmax\ p(c \mid x2) = p(c = 1\ |\{y1, y2, y3, y4\} = \{0.1, B, -0.1, -0.4\}) = 0.2610$ → false positive

$argmax\ p(c \mid x3) = p(c = 0\ |\{y1, y2, y3, y4\} = \{0.2, A, -0.1,0.2\}) = 0.2317$ → true negative

$argmax\, p(c \mid x4) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{0.1, C, 0.8{,}0.8\}) = 0.0831$ → false positive

$argmax\, p(c \mid x5) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{0.3, B, 0.1{,}0.3\}) = 0.2294$ → true positive

$argmax\, p(c \mid x6) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{-0.1, C, 0.2, -0.2\}) = 0.2430$ → true positive

$argmax\, p(c \mid x7) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{-0.3, C, -0.1{,}0.2\}) = 0.1207$ → true positive

$argmax\, p(c \mid x8) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{0.2, B, 0.5{,}0.6\}) = 0.2033$ → true positive

$argmax\, p(c \mid x9) = p(c = 0 \mid \{y1, y2, y3, y4\} = \{0.4, A, -0.4, -0.7\}) = 0.0598$ → false negative

$argmax\, p(c \mid x10) = p(c = 1 \mid \{y1, y2, y3, y4\} = \{-0.2, C, 0.4{,}0.3\}) = 0.3208$ → true positive

		TRUE/ACTUAL/TARGET	
EXPECTED		P	N
	P	5	2
	N	1	2

3) F-measure: $F = \frac{(\beta^2+1)PR}{\beta^2 P+R}$

F1 score: β=1 → $\frac{1}{F} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R})$

$Precision(P) = TPos/(TPos + FPos) = 5/7$

$Recall(R) = TPos/Pos = 5/6$

Therefore, $F = \frac{10}{13}$

4) Firstly, we calculate the posterior probabilities,

$P(Class = 0 \mid x1) = 0.8350241079996211$

$P(Class = 0 \mid x2) = 0.19507304046225138$

$P(Class = 0 \mid x3) = 0.7591425170623465$

$P(Class = 0 \mid x4) = 0.4587184648911188$

$P(Class = 0 \mid x5) = 0.4562537673157476$

$P(Class = 0 \mid x6) = 0.07245026327400551$

$P(Class = 0 \mid x7) = 0.06365986350782053$

$P(Class = 0 \mid x8) = 0.46651313999863186$

$P(Class = 0 \mid x9) = 0.6993428747731582$

$P(Class = 0 \mid x10) = 0.08637971094690976$

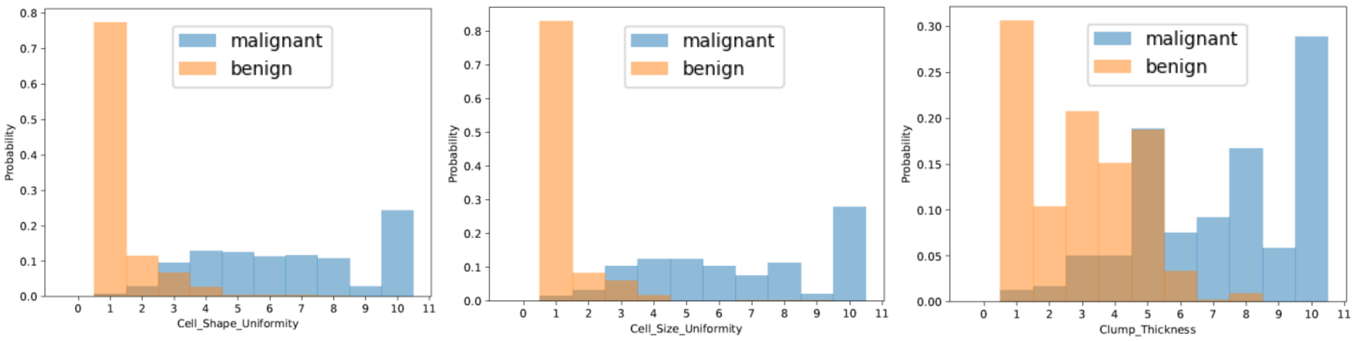
Secondly, we decide a set of thresholds against which to test our accuracy; those must include every single possible outcome for the accuracy, so there should be one threshold value between every two consecutive posterior probabilities.

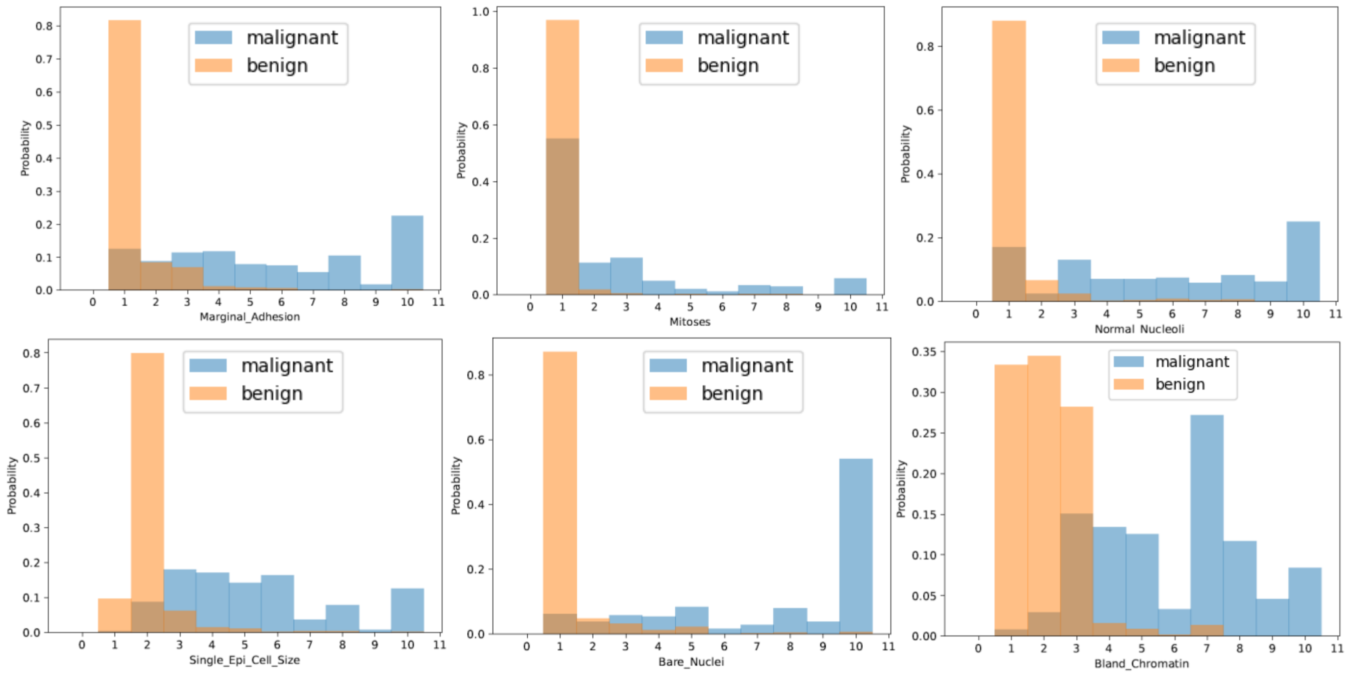
TRUE	Posterior Probability P(c=0|x)	0	0.07	0.08	0.100	0.300	0.457	0.460	0.600	0.700	0.800	1
0	0.8350	0	0	0	0	0	0	0	0	0	0	1
0	0.1951	0	0	0	0	1	1	1	1	1	1	1
0	0.7591	0	0	0	0	0	0	0	0	0	1	1
0	0.4587	0	0	0	0	0	0	1	1	1	1	1
1	0.4563	0	0	0	0	0	1	1	1	1	1	1
1	0.0725	0	0	1	1	1	1	1	1	1	1	1
1	0.0637	0	1	1	1	1	1	1	1	1	1	1
1	0.4665	0	0	0	0	0	0	0	1	1	1	1
1	0.6993	0	0	0	0	0	0	0	0	1	1	1
1	0.0864	0	0	0	1	1	1	1	1	1	1	1
Accuracy		4/10	5/10	6/10	7/10	6/10	7/10	6/10	7/10	8/10	7/10	

Therefore, one optimal decision probability threshold value is 0.7, with an accuracy of 8/10.
Any other value in]0.6993, 0.7591[would also be an optimal threshold.
This means there is a class imbalance between Negative and Positive such that much greater accuracy results are obtained from thresholds above 0.5. Consequently, we can say that the boundary separating our outcome classes is positively deviated. Also, the high accuracy (8/10) can mean a well-defined boundary between the two.

II. Programming and critical analysis

5)





6) Average *k*NN accuracies for:

k=3 → 0.9692668371696506

k=5 → 0.9722080136402388

k=7 → 0.9736786018755328

We obtained the best average accuracy across the 10-fold cross validation by using k=7, which means that this is the less susceptible value to overfitting risk, as a result of using more data for classification which smoothens the decision boundary and reduces noise pollution of the result.

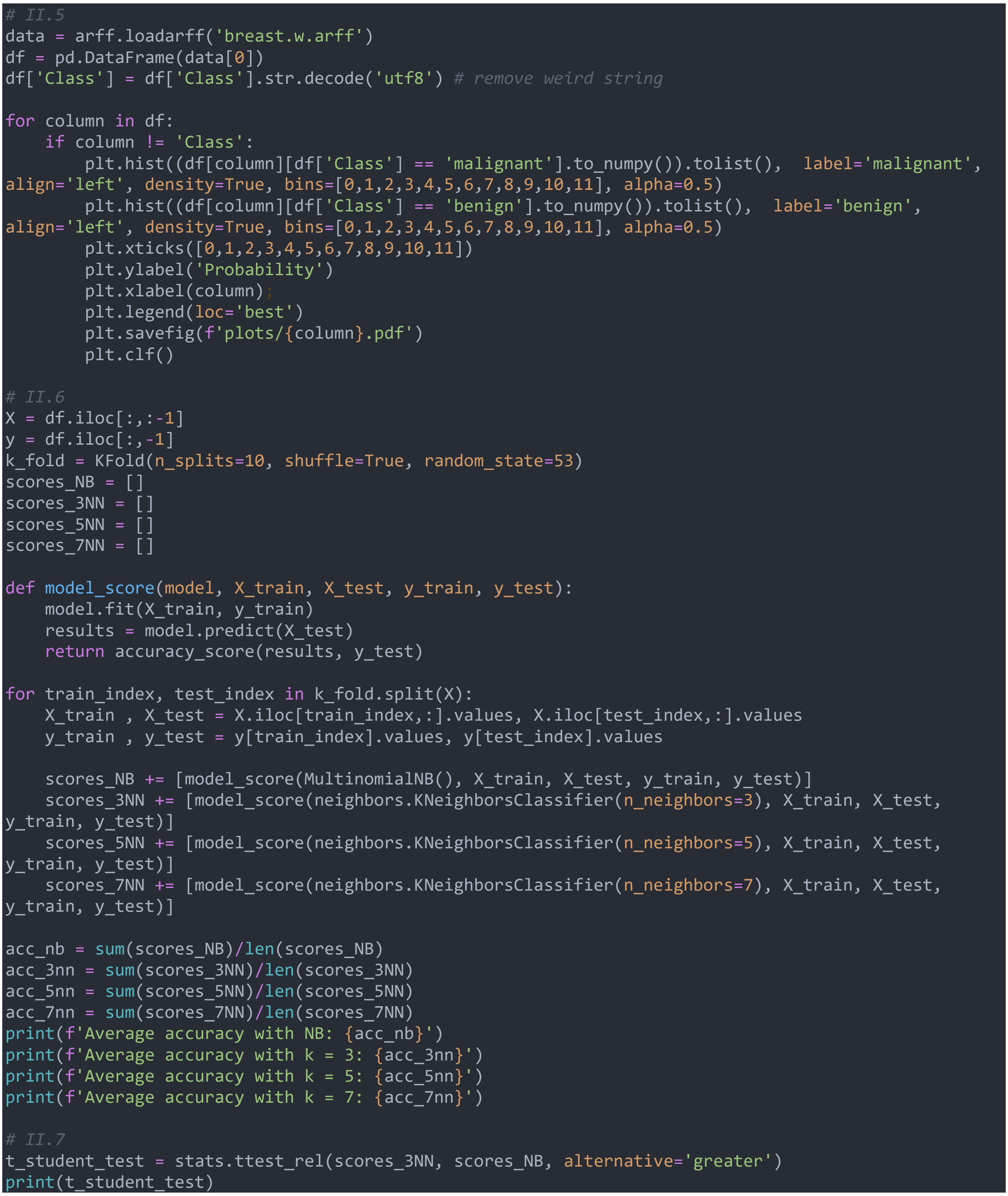
7) H0: *k*NN is statistically equivalent to Naïve Bayes (multinomial assumption)

H1: *k*NN is statistically superior to Naïve Bayes (right tail test)

Considering both models are used on the same dataset they are inherently related. Additionally, we can assume accuracy estimates are normally distributed. As a result, we used "*scipy.stats.ttest_rel*" function since its documentation matches our requirements. The values obtained were t-statistic=5.220199 and p-value=0.00027. p-value<<0.01 and t-statistic>2 meaning we can confidently reject the null hypothesis for the alternative one.

8) The explanatory variables are inevitably related in an individual's diagnosis so we should not assume independence between them. Therefore, the naïve bayes assumption leads to less-than-optimal results compared to the kNN classifier. Furthermore, not knowing the probability distributions for the input variables undermines the Naïve Bayes, resulting in poor predictions.

III. APPENDIX



```
# II.5
data = arff.loadarff('breast.w.arff')
df = pd.DataFrame(data[0])
df['Class'] = df['Class'].str.decode('utf8') # remove weird string

for column in df:
    if column != 'Class':
        plt.hist((df[column][df['Class'] == 'malignant'].to_numpy()).tolist(),  label='malignant',
align='left', density=True, bins=[0,1,2,3,4,5,6,7,8,9,10,11], alpha=0.5)
        plt.hist((df[column][df['Class'] == 'benign'].to_numpy()).tolist(),  label='benign',
align='left', density=True, bins=[0,1,2,3,4,5,6,7,8,9,10,11], alpha=0.5)
        plt.xticks([0,1,2,3,4,5,6,7,8,9,10,11])
        plt.ylabel('Probability')
        plt.xlabel(column);
        plt.legend(loc='best')
        plt.savefig(f'plots/{column}.pdf')
        plt.clf()

# II.6
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
k_fold = KFold(n_splits=10, shuffle=True, random_state=53)
scores_NB = []
scores_3NN = []
scores_5NN = []
scores_7NN = []

def model_score(model, X_train, X_test, y_train, y_test):
    model.fit(X_train, y_train)
    results = model.predict(X_test)
    return accuracy_score(results, y_test)

for train_index, test_index in k_fold.split(X):
    X_train , X_test = X.iloc[train_index,:].values, X.iloc[test_index,:].values
    y_train , y_test = y[train_index].values, y[test_index].values

    scores_NB += [model_score(MultinomialNB(), X_train, X_test, y_train, y_test)]
    scores_3NN += [model_score(neighbors.KNeighborsClassifier(n_neighbors=3), X_train, X_test,
y_train, y_test)]
    scores_5NN += [model_score(neighbors.KNeighborsClassifier(n_neighbors=5), X_train, X_test,
y_train, y_test)]
    scores_7NN += [model_score(neighbors.KNeighborsClassifier(n_neighbors=7), X_train, X_test,
y_train, y_test)]

acc_nb = sum(scores_NB)/len(scores_NB)
acc_3nn = sum(scores_3NN)/len(scores_3NN)
acc_5nn = sum(scores_5NN)/len(scores_5NN)
acc_7nn = sum(scores_7NN)/len(scores_7NN)
print(f'Average accuracy with NB: {acc_nb}')
print(f'Average accuracy with k = 3: {acc_3nn}')
print(f'Average accuracy with k = 5: {acc_5nn}')
print(f'Average accuracy with k = 7: {acc_7nn}')

# II.7
t_student_test = stats.ttest_rel(scores_3NN, scores_NB, alternative='greater')
print(t_student_test)
```

END