I. Pen-and-paper

1)

Homework II

I. Pen and Paper

1. -

Observations used for classification

	$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	$x^{(6)}$	$x^{(7)}$	$x^{(8)}$	distance Weighted kNN (k=5) Computation for each class		predicted classif.
vector features	$\binom{A}{0}$	$\binom{B}{1}$	$\binom{A}{1}$	$\binom{A}{0}$	$\binom{B}{0}$	$\binom{B}{0}$	$\binom{A}{1}$	$\binom{B}{1}$			
actual classif.	P	P	P	P	N	N	N	N			
$x^{(1)}$	—	2.5	1.5	0.5	1.5	1.5	1.5	2.5	P: $\frac{1}{1.5}+\frac{1}{0.5}=2.\overline{6}$	N: $\frac{1}{1.5}+\frac{1}{1.5}+\frac{1}{1.5}=2$	P (TP)
$x^{(2)}$	2.5	—	1.5	2.5	1.5	1.5	1.5	0.5	P: $\frac{1}{1.5}=0.\overline{6}$	N: $3\times\frac{1}{1.5}+\frac{1}{0.5}=4$	N (FN)
$x^{(3)}$	1.5	1.5	—	1.5	2.5	2.5	0.5	1.5	P: $3\times\frac{1}{1.5}=2$	N: $\frac{1}{1.5}+\frac{1}{0.5}=2.\overline{6}$	N (FN)
$x^{(4)}$	0.5	2.5	1.5	—	1.5	1.5	1.5	2.5	P: $\frac{1}{0.5}+\frac{1}{1.5}=2.\overline{6}$	N: $3\times\frac{1}{1.5}=2$	P (TP)
$x^{(5)}$	1.5	1.5	2.5	1.5	—	0.5	2.5	1.5	P: $3\times\frac{1}{1.5}=2$	N: $\frac{1}{0.5}+\frac{1}{1.5}=2.\overline{6}$	N (TN)
$x^{(6)}$	1.5	1.5	2.5	1.5	0.5	—	2.5	1.5	P: $3\times\frac{1}{1.5}=2$	N: $\frac{1}{0.5}+\frac{1}{1.5}=2.\overline{6}$	N (TN)
$x^{(7)}$	1.5	1.5	0.5	1.5	2.5	2.5	—	1.5	P: $3\times\frac{1}{1.5}+\frac{1}{0.5}=4$	N: $\frac{1}{1.5}=0.\overline{6}$	P (FP)
$x^{(8)}$	2.5	0.5	1.5	2.5	1.5	1.5	1.5	—	P: $\frac{1}{0.5}+\frac{1}{1.5}=2.\overline{6}$	N: $3\times\frac{1}{1.5}=2$	P (FP)

query vectors: $x^{(1)}$ … $x^{(8)}$

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{2}{2+2} = \frac{1}{2}$$

FN = 2 TP = 2

2.- $P(C=N|y_1,y_2,y_3) = \dfrac{P(y_1,y_2,y_3|C=N)\,P(C=N)}{P(y_1,y_2,y_3)}$

$\boxed{P(C=N)=\frac{4}{9} \quad P(C=P)=\frac{5}{9}}$

$= \dfrac{P(y_1,y_2|C=N)\times P(y_3|C=N)\times P(C=N)}{P(y_1,y_2,y_3|C=N)P(C=N) + P(y_1,y_2,y_3|C=P)P(C=P)} =$

$= \dfrac{\frac{4}{9}P(y_1,y_2|C=N)\times P(y_3|C=N)}{\frac{4}{9}P(y_1,y_2|C=N)P(y_3|C=N) + \frac{5}{9}P(y_1,y_2|C=P)\times P(y_3|C=P)}$ (1)

$P(y_3|C=N) \simeq \mathcal{N}(y_3|\mu_N,\sigma_N^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma_N^2}(y_3-\mu_N)^2\right)$ (2)

Where the maximum likelihood parameters of the Gaussian are:

$\mu_N = \frac{1+0.9+1.2+0.8}{4} = 0.975 \quad \sigma^2 = \frac{(1-0.975)^2+(0.9-0.975)^2+(1.2-0.975)^2+(0.8-0.975)^2}{3} = 0.0292$

$P(y_3|C=P) \simeq \mathcal{N}(y_3|\mu_P,\sigma_P^2) = \frac{1}{\sqrt{2\pi}\sigma_P}\exp\left(-\frac{1}{2\sigma_P^2}(y_3-\mu_P)^2\right)$ (3)

Where the maximum likelihood parameters of the Gaussian are:

$\mu_P = \frac{1.2+0.8+0.5+0.9+0.8}{5} = 0.840$

$\sigma_P^2 = \frac{(1.2-0.840)^2+(0.8-0.840)^2+(0.5-0.840)^2+(0.4-0.840)^2+(0.8-0.840)^2}{4}$

$= 0.0630$

The joint distributions are:

$\boxed{P(y_1,y_2|C=N)}$

y_1/y_2	0	1
A	0	$\frac{1}{4}$
B	$\frac{2}{4}=\frac{1}{2}$	$\frac{1}{4}$

$\boxed{P(y_1,y_2|C=P)}$

y_1/y_2	0	1
A	$\frac{2}{5}$	$\frac{1}{5}$
B	$\frac{1}{5}$	$\frac{1}{5}$

Finally, given generic parameters y_1, y_2, y_3 and using eqs. (1), (2), (3) and the joint distribution then we can calculate $P(C=N \mid y_1, y_2, y_3)$:

y_1/y_2	0	1
A	0	$\dfrac{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)+\frac{5}{9}\times\frac{1}{5}\times\mathcal{N}(y_3 \mid \mu=0.840, \sigma^2=0.063)}$
B	$\dfrac{\frac{4}{9}\times\frac{1}{2}\times\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9}\times\frac{1}{2}\times\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)+\frac{5}{9}\times\frac{1}{5}\times\mathcal{N}(y_3 \mid \mu=0.840, \sigma^2=0.0630)}$	$\dfrac{\frac{4}{9}\times\frac{1}{4}\,\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(y_3 \mid \mu=0.975, \sigma^2=0.0292)+\frac{5}{9}\times\frac{1}{5}\times\mathcal{N}(y_3 \mid \mu=0.840, \sigma^2=0.0630)}$

And, $P(C=P \mid y_1, y_2, y_3) = 1 - P(C=N \mid y_1, y_2, y_3)$ (4)

since $P \cup N = \emptyset$ and $P \cap N = \Omega$

Then choose the class for which the posterior for the given parameters is larger (under MAP assumption).

3.- Under MAP assumption, given only two classes if $P(C=P \mid \vec{x}) > 0.5$ this means $1 - P(C=N \mid \vec{x}) \geq 0.5 \Rightarrow P(C=N \mid \vec{x}) < 0.5$ so we choose class positive, otherwise choose negative.

To compute the posterior just use eq. (4) and the table calculated in exercise 2.:

a) $\vec{x}^{(1)} = \begin{pmatrix} A \\ 1 \\ 0.8 \end{pmatrix}$

$$P(C=P \mid \vec{x}) = 1 - P(C=N \mid y_1=A, y_2=1, y_3=0.8)$$

$$= 1 - \frac{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(0.8 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(0.8 \mid \mu=0.975, \sigma^2=0.0292)+\frac{5}{9}\times\frac{1}{5}\times\mathcal{N}(0.8 \mid \mu=0.84, \sigma^2=0.063)}$$

$$= 0.5318$$

Under MAP assumption choose class positive

b) $\vec{x}^{(2)} = \begin{pmatrix} B \\ 1 \\ 1 \end{pmatrix}$

$$P(C=P \mid \vec{x}) = 1 - P(C=N \mid y_1=B, y_2=1, y_3=1)$$

$$= 1 - \frac{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(1 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9}\times\frac{1}{4}\times\mathcal{N}(1 \mid \mu=0.975, \sigma^2=0.0292)+\frac{5}{9}\times\frac{1}{5}\times\mathcal{N}(1 \mid \mu=0.84, \sigma^2=0.063)}$$

$$= 0.3596$$

Under MAP assumption choose class negative

c) $\vec{x}^{(3)} = \begin{pmatrix} B \\ 0 \\ 0.9 \end{pmatrix}$ $P(c=P \mid \vec{x}) = 1 - P(c=N \mid y_1=B, y_2=0, y_3=0.9)$

$$= 1 - \frac{\frac{4}{9} \times \frac{1}{2} \times \mathcal{N}(0.9 \mid \mu=0.975, \sigma^2=0.0292)}{\frac{4}{9} \times \frac{1}{2} \times \mathcal{N}(0.9 \mid \mu=0.975, \sigma^2=0.0292) + \frac{5}{9} \times \frac{1}{5} \times \mathcal{N}(0.9 \mid \mu=0.84, \sigma^2=0.063)}$$

$= 0.2670 \rightarrow$ $\boxed{\text{Under MAP assumption choose class N}}$

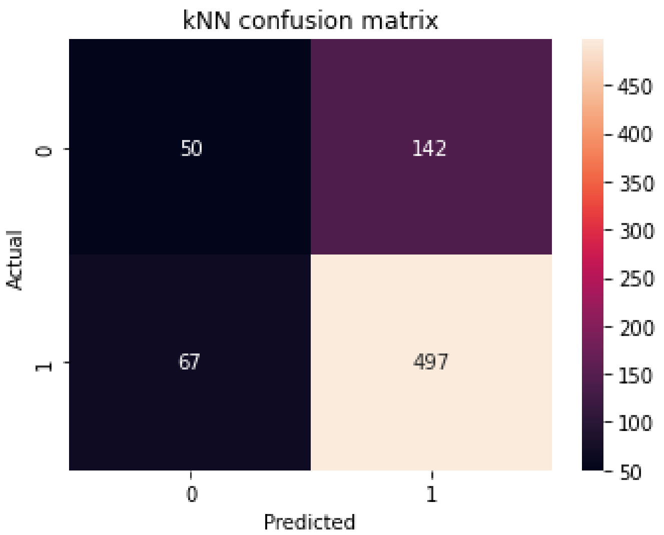
4.-

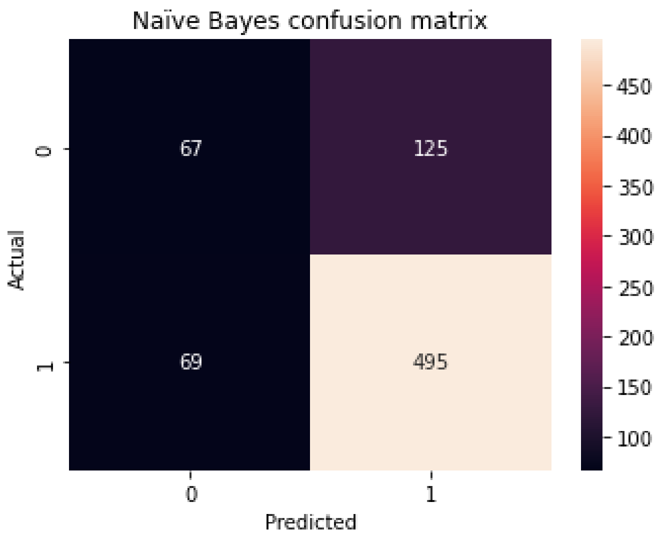
Actual class →	Positive	Positive	Negative	
$\theta / \vec{x}$	$\vec{x}^{(1)} = \begin{pmatrix} A \\ 1 \\ 0.8 \end{pmatrix}$	$\vec{x}^{(2)} = \begin{pmatrix} B \\ 1 \\ 1 \end{pmatrix}$	$\vec{x}^{(3)} = \begin{pmatrix} B \\ 0 \\ 0.9 \end{pmatrix}$	
0.3	$f(\vec{x}^{(1)} \mid \theta) = P$ (TP)	$f(\vec{x}^{(2)} \mid \theta) = P$ (TP)	$f(\vec{x}^{(3)} \mid \theta) = N$ (TN)	Accuracy $= \frac{TP+TN}{All} = \frac{2+1}{3} = 1$
0.5	$f(\vec{x}^{(1)} \mid \theta) = P$ (TP)	$f(\vec{x}^{(2)} \mid \theta) = N$ (FN)	$f(\vec{x}^{(3)} \mid \theta) = N$ (TN)	Accuracy $= \frac{1+1}{3} = \frac{2}{3}$
0.7	$f(\vec{x}^{(1)} \mid \theta) = N$ (FN)	$f(\vec{x}^{(2)} \mid \theta) = N$ (FN)	$f(\vec{x}^{(3)} \mid \theta) = N$ (TN)	Accuracy $= \frac{0+1}{3} = \frac{1}{3}$
	$P(c=P \mid \vec{x}^{(1)}) = 0.5318$	$P(c=P \mid \vec{x}^{(2)}) = 0.3596$	$P(c=P \mid \vec{x}^{(3)}) = 0.2670$	

So the treshold $\theta = 0.3$ optimizes testing accuracy

II. Programming and critical analysis

5) The cumulative testing confusion matrices of kNN and Naïve Bayes:





6) Given the obtained p-value of 0.9104, which is very high, we cannot reject the null hypotheses of identical average scores.
In fact, if we run the test with hypothesis "*k*NN is statistically inferior to Naïve Bayes regarding accuracy", we get a p-value of 0.0896, which means we can accept that hypothesis depending on the chosen treshold (a treshold of 0.09 is perfectly reasonable while a treshold of 0.92 is ludicrous).

7) Analysis as to why Naïve Bayes may be better (according to the t-test) at classifying the test data:

- Interestingly if we uncomment lines 30 and 31, thereby normalizing the data before using the classifiers, we obtain a p-value of 0.0013 for the hypothesis "*k*NN is statistically superior to Naïve Bayes regarding accuracy" which means we can reject the null hypothesis that the accuracy scores are identical and convincingly accept the hypothesis that "*k*NN is statistically superior to Naïve Bayes regarding accuracy". So one reason for the observed differences in predictive accuracy is that the data is not normalized which penalizes kNN much more than Naïve Bayes, since certain features will have too much influence when computing the distances.
- Secondly, the high-dimensionality of the data is also highly penalizing for the kNN because this classifier measures distances between points thereby suffering greatly from the curse of dimensionality. This doesn't affect Naïve Bayes so much since this classifier treats all features as being independent of each other.

III. APPENDIX

```
1. from scipy.io.arff import loadarff
2. import pandas as pd
3. import numpy as np
4. from sklearn.neighbors import KNeighborsClassifier
5. from sklearn.naive_bayes import GaussianNB
6. from sklearn.model_selection import StratifiedKFold
7. from sklearn.metrics import confusion_matrix, accuracy_score
8. from sklearn.preprocessing import StandardScaler
9. from scipy import stats
10.import seaborn as sbn
11.import matplotlib.pyplot as plt
12.
13.data = loadarff('pd_speech.arff')
14.df = pd.DataFrame(data[0])
15.df['class'] = df['class'].str.decode('utf-8')
16.x, y = df.drop('class', axis=1), np.ravel(df['class'])
17.
18.folds = StratifiedKFold(n_splits=10, shuffle=True, random_state=0)
19.knn_predictor = KNeighborsClassifier() #by default it uses uniform weights, k=5 and
   euclidean distance
20.gnb_predictor = GaussianNB()
21.
22.knn_confusion_mtrx = np.zeros((2,2))
```

```
knn_accs = [] #accuracies of each fold for the knn_predictor
gnb_confusion_mtrx = np.zeros((2,2))
gnb_accs = [] #accuracies of each fold for the gnb_predictor

for train_index, test_index in folds.split(x, y):
    x_train, x_test = x.iloc[train_index], x.iloc[test_index]
    y_train, y_test = y[train_index], y[test_index]
    #scaler = StandardScaler().fit(x_train)
    #x_train, x_test = scaler.transform(x_train), scaler.transform(x_test)

    #kNN
    knn_predictor.fit(x_train, y_train)
    knn_y_pred = knn_predictor.predict(x_test)
    knn_confusion_mtrx = knn_confusion_mtrx + confusion_matrix(y_test, knn_y_pred)
    knn_accs.append(accuracy_score(y_test, knn_y_pred))

    #Naive Bayes
    gnb_predictor.fit(x_train, y_train)
    gnb_y_pred = gnb_predictor.predict(x_test)
    gnb_confusion_mtrx = gnb_confusion_mtrx + confusion_matrix(y_test, gnb_y_pred)
    gnb_accs.append(accuracy_score(y_test, gnb_y_pred))

fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(12, 4))
sbn.heatmap(knn_confusion_mtrx, annot=True, fmt='g', ax=ax1).set(xlabel='Predicted', ylabel='Actual')
ax1.set_title("kNN confusion matrix")
sbn.heatmap(gnb_confusion_mtrx, annot=True, fmt='g', ax=ax2).set(xlabel='Predicted', ylabel='Actual')
ax2.set_title("Naive Bayes confusion matrix")
plt.show()

print(stats.ttest_rel(knn_accs, gnb_accs, alternative='greater'))
print(stats.ttest_rel(knn_accs, gnb_accs, alternative='less'))
```

END