ENGR 421/DASC 521: Introduction to Machine Learning
Homework 1: Naive Bayes Classifier
Deadline: March 17, 2025, 11:59 PM

In this homework, you will implement a naive Bayes classifier using Python. Here are the steps you need to follow:

1. Read Section 5.7 from the textbook.

2. You are given a multivariate classification data set, which contains 11314 and 7532 documents. Each document is represented as a 2000-dimensional binary vector. Each feature shows whether a word appears in the corresponding document or not. These documents are from 20 distinct classes, namely, 1, 2, ..., 20. You are provided with four data files:

   a. `20newsgroup_words_train.csv`: training documents,

   b. `20newsgroup_labels_train.csv`: training labels,

   c. `20newsgroup_words_test.csv`: test documents,

   d. `20newsgroup_labels_test.csv`: test labels.

3. Calculate the prior probability estimates $\widehat{\Pr}(y = 1), \widehat{\Pr}(y = 2), \ldots, \widehat{\Pr}(y = 20)$ using the data points in the training set. (20 points)

```
class_priors = estimate_prior_probabilities(y_train)
print(class_priors)

[0.04242531 0.05161747 0.05223617 0.05214778 0.05108715 0.05241294
 0.05170585 0.05250133 0.05285487 0.05276648 0.05303164 0.05258971
 0.05223617 0.05250133 0.05241294 0.05294326 0.04825879 0.04984974
 0.04109952 0.03332155]
```

Hint: You can use the following equation to calculate the prior probability estimates.

$$\widehat{\Pr}(y = c) = \frac{\sum_{i=1}^{N} 1(y_i = c)}{N} = \frac{N_c}{N}$$

Hint: Let us define the following probability.

$$\pi_{cd} = \text{probability of having word } d \text{ for a document in class } c.$$

---

4. Calculate the model parameter estimates $\hat{\pi}_{1,1}, \hat{\pi}_{1,2}, \ldots, \hat{\pi}_{1,2000},\ \hat{\pi}_{2,1}, \hat{\pi}_{2,2}, \ldots, \hat{\pi}_{2,2000},\ \ldots, \hat{\pi}_{20,1}, \hat{\pi}_{20,2}, \ldots, \hat{\pi}_{20,2000}$ using the data points in the training set. (20 points)

```
P = estimate_success_probabilities(X_train, y_train)
print(P)

[[3.63636364e-03 1.27272727e-02 1.38636364e-02 ... 2.65000000e-01
```

```
 1.38636364e-02 2.75000000e-02]
[2.05284553e-02 1.13821138e-02 1.03658537e-02 ... 9.67479675e-02
 6.30081301e-03 7.31707317e-03]
[1.93743693e-02 1.02926337e-02 8.27447023e-03 ... 1.19273461e-01
 2.01816347e-04 5.24722503e-03]
...
[3.02904564e-02 8.73443983e-02 2.28215768e-03 ... 2.50207469e-01
 2.92531120e-02 3.54771784e-02]
[1.87283237e-02 3.49132948e-02 2.54335260e-03 ... 2.03699422e-01
 1.17919075e-02 1.98843931e-02]
[5.40540541e-03 9.26640927e-03 7.97940798e-03 ... 1.99742600e-01
 1.57014157e-02 4.01544402e-02]]
```

Hint: You can use the following equation to calculate the model parameter estimates

$$\hat{\pi}_{cd} = \frac{\sum_{i=1}^{N} x_{id} 1(y_i = c) + \alpha}{N_c + \alpha D}$$

where you need to add α to the numerator and αD to the denominator to avoid 0 probabilities, which is known as Laplace smoothing. Please set α to 0.2.

---

5. Calculate the score values for the data points in training and test sets using the estimated parameters. (40 points)

```
scores_train = calculate_score_values(X_train, P, class_priors)
print(scores_train)
```

```
[[-247.57307095 -246.03514443 -246.96022074 ... -245.59860283
  -250.8865859  -243.5830906 ]
 [-259.06090578 -214.30799791 -217.75760323 ... -245.96994278
  -244.58737797 -250.76752986]
 [-501.73003729 -490.94595825 -486.21776519 ... -482.12640496
  -468.75238602 -503.38719951]
 ...
 [-228.78523136 -214.17362279 -211.81253918 ... -240.82316635
  -230.50860465 -225.98831035]
 [-282.15638893 -259.57033559 -277.65308446 ... -277.3058842
  -283.94370355 -293.32449436]
 [-175.8187118  -167.08927009 -171.54533879 ... -213.60147007
  -190.97151986 -193.80063856]]
```

```
scores_test = calculate_score_values(X_test, P, class_priors)
print(scores_test)
```

```
[[-250.38534464 -229.26402493 -232.99579397 ... -258.0922156
  -245.28145541 -246.4353936 ]
 [-236.6117014  -226.16280559 -226.38909216 ... -243.30176501
  -237.40318505 -244.62522108]
```

```
[-138.6482086  -148.45431847 -152.64766278 ... -179.56793033
 -167.633637   -150.07549428]
...
[-299.1732706  -333.05131454 -326.84794361 ... -306.62141197
 -293.34419338 -302.75465874]
[-280.10753299 -235.77479771 -234.78743911 ... -276.21669496
 -278.12803044 -268.37893615]
[-290.77215093 -333.2387222  -334.19348471 ... -302.75008046
 -293.47750156 -295.77620516]]
```

Hint: You can use the following equation to calculate the score values.

$$
\begin{aligned}
g_c(\boldsymbol{x}) &= \log\left[\prod_{d=1}^{D} \hat{p}(x_d|y=c)\right] + \log \widehat{\Pr}(y=c) \\
&= \log\left[\prod_{d=1}^{D} \left(\hat{\pi}_{cd}^{x_d}(1-\hat{\pi}_{cd})^{(1-x_d)}\right)\right] + \log \widehat{\Pr}(y=c)
\end{aligned}
$$

---

6. Calculate the confusion matrices for the data points in training and test sets using the calculated score values. (20 points)

```
Training accuracy is 78.10%.
Test accuracy is 60.74%.
```

What to submit: You need to submit your source code in a single file (`.py` file). You are provided with a template file named as `0099999.py`, where `99999` should be replaced with your 5-digit student number. You are allowed to change the template file between the following lines.

```
# your implementation starts below

# your implementation ends above
```

How to submit: Submit the file you edited to LearnHub by following the exact style mentioned. Submissions that do not follow these guidelines will not be graded.

Late submission policy: Late submissions will not be graded.

Cheating policy: Very similar submissions will not be graded.
