ENGR 421/DASC 521: Introduction to Machine Learning
Homework 2: Multivariate Parametric Classification
Deadline: March 25, 2024, 11:59 PM

In this homework, you will implement a multivariate parametric classification using Python. Here are the steps you need to follow:

1. Read Chapter 5 from the textbook.
2. You are given a multivariate classification data set, which contains 4000 data points from a two-dimensional feature space. These data points are from four distinct classes, where we have 1000 data points from each class. You are provided with two data files:
   a. `hw02_data_points.csv`: two-dimensional data points,
   b. `hw02_class_labels.csv`: corresponding class labels.
3. Calculate the prior probability estimates $\widehat{\Pr}(y = 1)$, $\widehat{\Pr}(y = 2)$, $\widehat{\Pr}(y = 3)$, and $\widehat{\Pr}(y = 4)$ using the training data points. (10 points)

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

[0.25 0.25 0.25 0.25]
```

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

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

---

4. Calculate the class mean estimates $\hat{\boldsymbol{\mu}}_1$, $\hat{\boldsymbol{\mu}}_2$, $\hat{\boldsymbol{\mu}}_3$, and $\hat{\boldsymbol{\mu}}_4$ using the training data points. (20 points)

```
sample_means = estimate_class_means(X_train, y_train)
print(sample_means)

[[ -7.48177328 -43.30108951]
 [-32.54451927   3.0991768 ]
 [ -0.98957165  37.36564372]
 [ 41.0158642    2.83626898]]
```

Hint: You can use the following equation to calculate the class mean estimates.

$$\hat{\boldsymbol{\mu}}_c = \frac{\sum\limits_{i=1}^{N} \boldsymbol{x}_i 1(y_i = c)}{\sum\limits_{i=1}^{N} 1(y_i = c)}$$

---

5. Calculate the class covariance estimates $\hat{\mathbf{\Sigma}}_1$, $\hat{\mathbf{\Sigma}}_2$, $\hat{\mathbf{\Sigma}}_3$, and $\hat{\mathbf{\Sigma}}_4$ using the training data points. (20 points)

```
sample_covariances = estimate_class_covariances(X_train, y_train)
print(sample_covariances)
```

```
[[[ 595.91795322  -97.28384176]
  [ -97.28384176   70.0541601 ]]

 [[ 228.83361511 -106.71382252]
  [-106.71382252  421.73601334]]

 [[ 436.41712588  148.46104888]
  [ 148.46104888  322.51670977]]

 [[ 247.81399618 -175.44912866]
  [-175.44912866  378.82553071]]]
```

Hint: You can use the following equation to calculate the class covariance estimates.

$$\hat{\mathbf{\Sigma}}_c = \frac{\sum\limits_{i=1}^{N}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_c)^\top 1(y_i = c)}{\sum\limits_{i=1}^{N} 1(y_i = c)}$$

---

6. Calculate the score values for the data points in your training set using the estimated parameters. (30 points)

```
scores_train = calculate_score_values(X_train, sample_means,
                                      sample_covariances, class_priors)
print(scores_train)
```

```
[[ -8.83707237 -12.13230093 -21.45276289 -31.0933647 ]
 [ -8.90619198 -12.08434508 -21.29056001 -32.14774969]
 [ -8.8635333  -14.0070853  -24.39646128 -23.1262415 ]
 ...
 [-57.2734888  -28.1751339  -14.80342019  -9.93289617]
 [-59.63926912 -27.5650189  -14.13883705  -9.91693716]
 [-57.60340331 -28.96744342 -15.14833814 -10.11913532]]
```

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

$$\begin{aligned} g_c(\boldsymbol{x}) &= \log \hat{p}(\boldsymbol{x}|y = c) + \log \widehat{\Pr}(y = c) \\ &= -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log(|\hat{\mathbf{\Sigma}}_c|) - \frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_c)^\top \hat{\mathbf{\Sigma}}_c^{-1}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_c) + \log \widehat{\Pr}(y = c) \end{aligned}$$

---

7. Calculate the confusion matrix for the training data points using the calculated score values. (10 points)

```
confusion_train = calculate_confusion_matrix(y_train, scores_train)
print(confusion_train)

[[991 141   0   0]
 [  9 800 155   0]
 [  0  59 754 136]
 [  0   0  91 864]]
```

8. Calculate the shared covariance estimate $\hat{\boldsymbol{\Sigma}}_1 = \hat{\boldsymbol{\Sigma}}_2 = \hat{\boldsymbol{\Sigma}}_3 = \hat{\boldsymbol{\Sigma}}_4 = \hat{\boldsymbol{\Sigma}}$ using the training data points. (10 points)

```
sample_covariances = estimate_shared_class_covariance(X_train, y_train)
print(sample_covariances)

[[[1076.8464311    17.86950334]
  [  17.86950334 1120.48935404]]

 [[1076.8464311    17.86950334]
  [  17.86950334 1120.48935404]]

 [[1076.8464311    17.86950334]
  [  17.86950334 1120.48935404]]

 [[1076.8464311    17.86950334]
  [  17.86950334 1120.48935404]]]

scores_train = calculate_score_values(X_train, sample_means,
                                      sample_covariances, class_priors)
print(scores_train)

[[-10.25994863 -11.58543347 -13.59080705 -12.79091744]
 [-10.27611812 -11.54690246 -13.61489931 -12.90377425]
 [-10.34886152 -12.28681304 -13.75814064 -12.04234436]
 ...
 [-13.44098936 -13.57539411 -11.6897625  -10.37141859]
 [-13.45097838 -13.43588688 -11.54405212 -10.38556672]
 [-13.53278397 -13.72024542 -11.78663794 -10.38815203]]

confusion_train = calculate_confusion_matrix(y_train, scores_train)
print(confusion_train)

[[924  88   0   0]
 [ 76 883 163   0]
 [  0  29 810 168]
 [  0   0  27 832]]
```

Hint: You can use the following equations to calculate the shared covariance estimate.

$$\hat{\boldsymbol{\mu}} = \frac{\sum\limits_{i=1}^{N} \boldsymbol{x}_i}{N}$$

$$\hat{\boldsymbol{\Sigma}}_1 = \hat{\boldsymbol{\Sigma}}_2 = \hat{\boldsymbol{\Sigma}}_3 = \hat{\boldsymbol{\Sigma}}_4 = \hat{\boldsymbol{\Sigma}} = \frac{\sum\limits_{i=1}^{N} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^\top}{N}$$

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 Blackboard 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.