Assignment 2 - Q1

Question 1: MMD Classifier (20 pts)

Let's consider the following two classes:

- Class C_1: $x_1, x_2 \sim \mathcal{N}(\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix})$
- Class C_2: $x_1, x_2 \sim \mathcal{N}(\mu = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix})$

1. Code the two-class multivariate normal distribution (mean 2 samples each) for the above two classes, and plot the samples. Use the following random seed: `np.random.seed(0)`. Use the colors red and blue for C_1 and C_2 respectively in the data samples. For example, let's let C_1 be a 2-sample generated by $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and let C_2 be a 2-sample generated by $\mu = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Plot the 2-sample samples of C_1 in red and the 2-sample samples of C_2 in blue. Use the random seed provided above for the two classes. Use the 2-sample samples for the two classes to learn the MMD classifier.

   Recall that the MMD classifier is a classifier that is based on the expectation and dispersion of the underlying distribution for each class. Use the expectation and dispersion to set the centers of the Gaussian kernels.

2. Using the above calculations, find the decision boundary for the MMD classifier by hand. Plot the boundary and the data.

3. Plot Using the above calculations, find the decision boundary for the MMD classifier by hand. Plot the boundary and the data.

4. Code the MMD classifier using scikit-learn and plot the decision boundary for the two classes. Use the kernel bandwidth for the MMD classifier as 0.25(sec. Plot the boundary and the data.

5. Compare the MMD classifier with the Gaussian Naive Bayes classifier. Which classifier is better? Why? Use the following random seed for the two classes: `np.random.seed(1)`. Use the 2-sample samples for the two classes (50 samples per class). Report the classification accuracy of the classifiers using:

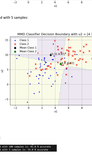
   `from sklearn.metrics import accuracy_score`
   `accuracy_score(y_true, y_pred)`

   Decision Boundary for samples with white noise:

   For mean = [4 5]

   MMD model trained with 100 samples:

   MMD model trained with 3 samples:

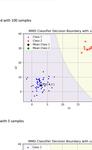


   Reported accuracy:

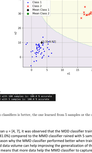
```
The MMD model trained with 100 samples has a 0.0 accuracy
The MMD model trained with 3 samples has a 0.0 accuracy
```

   For mean = [20 35]

   MMD Model trained with 100 samples



   MMD Model trained with 3 samples



   Reported accuracy:

```
The MMD model trained with 100 samples has a 0.0 accuracy
The MMD model trained with 3 samples has a 0.0 accuracy
```

6. (3 pts) Now, compare the MMD classifier in step 5 with the MDD classifier that you developed in the previous assignment. You will need to run the MDD on the data generated in this assignment for a fair comparison. Is MMD better than MDD? Explain.

```
The MMD model trained with 100 samples has a 0.0 accuracy
The MDD model trained with 100 samples has a 0.0 accuracy
The MMD model trained with 3 samples has a 0.0 accuracy
The MDD model trained with 3 samples has a 0.0 accuracy
```

As observed in the accuracy values, the MMD kind of a higher accuracy than the MDD classifier, for both training data (3 and 100 samples). This is suggested since the MMD classifier focuses on measuring the distance between probability distributions, helping in the detection and classification. This happens since the MMD uses the inverse covariance matrix which helps encoding the relationships between the features in the dataset. The inverse covariance helps to determine feature relationships and how the variables are dependent on each other.

7. (3 pts) Which of the two classifiers is better, the one trained from 3 samples or the one with 100 samples? Why?

When testing with the mean = [4, 5], it was observed that the MDD classifier trained with 100 samples achieved a higher accuracy than the MMD classifier trained with 3 samples. The MDD achieved an accuracy of 92.0%. The reason why the MMD classifier performed better when trained with more data is that having an increased data volume can help improving the generalization of the data and the performance of the model. The MDD classifier may have better performance with more data because it uses data distribution. Also, having more data reduces the variance in the estimated distributions, leading to more accurate predictions.

Moreover, more training data can help the classifier to have a better distinction of different classes and probabilities distributions.

8. (2 pts) Now, compare the MMD classifier in step 5 with the MDD classifier that you developed in the previous assignment. You will need to run the MDD on the data generated in this assignment for a fair comparison. Is MMD better than MDD? Explain.

```
The MMD model trained with 100 samples has a 0.0 accuracy
The MDD model trained with 100 samples has a 0.0 accuracy
The MMD model trained with 3 samples has a 0.0 accuracy
The MDD model trained with 3 samples has a 0.0 accuracy
```

As observed in the accuracy values, the MMD kind of a higher accuracy than the MDD classifier, for both training data (3 and 100 samples). This is suggested since the MMD classifier focuses on measuring the distance between probability distributions, helping in the detection and classification. This happens since the MMD uses the inverse covariance matrix which helps encoding the relationships between the features in the dataset. The inverse covariance helps to determine feature relationships and how the variables are dependent on each other.

Assignment 2 - Q2

October 17, 2023 12:40 PM

In this exercise, you will use the same data as in exercise 1 for the ML and MAP classifiers.

1. (5 pts) Using the sample mean and covariance calculated for the 100 sample data of the two classes (step 4 of exercise 1), you can estimate the probability distributions of the two classes. Based on this, develop an ML classifier and test it on the data generated in step 5 of exercise 1. Report the classification accuracy.

$$\phi(\mathbf{x}) = \left(\frac{1}{2\pi}\right)^{p/2} |\Sigma|^{-1/2} \exp\{-\frac{1}{2}(\mathbf{x}-\mu)'\Sigma^{-1}(\mathbf{x}-\mu)\}$$

```
The probabily distribution of class 1 is:  [[0.00507023 0.01600421 0.0082976  ... 0.01387914 0.0157202  0.00759167]
 [0.01600421 0.00346958 0.01485291 ... 0.01280392 0.01284301 0.00551071]
 [0.0082976  0.01485291 0.01128907 ... 0.01546017 0.01674874 0.00965249]
 ...
 [0.01387914 0.01280392 0.01546017 ... 0.0170072  0.01757182 0.01184851]
 [0.0157202  0.01284301 0.01674874 ... 0.01757182 0.01795716 0.01277116]
 [0.00759167 0.00551071 0.00965249 ... 0.01184851 0.01277116 0.00480859]]
The probabily distribution of class 2 is:  [[0.01297803 0.02040532 0.01197041 ... 0.02044678 0.04844309 0.03571493]
 [0.02040532 0.00834938 0.01026861 ... 0.01871291 0.00939277 0.00604582]
 [0.01197041 0.01026861 0.00682894 ... 0.01803891 0.02726342 0.0146394 ]
 ...
 [0.02044678 0.01871291 0.01803891 ... 0.02141423 0.02141818 0.01910773]
 [0.04844309 0.00939277 0.02726342 ... 0.02141818 0.00265707 0.00275441]
 [0.03571493 0.00604582 0.0146394  ... 0.01910773 0.00275441 0.00205496]]
```

The accuracy:

```
The accuracy of the ML classifier is:  73.0 %
```

2. (5 pts) Now, let's assume that the prior probabilities for the two classes are $p(C_1) = 0.58$ and $p(C_2) = 0.42$. Using these prior probabilities, and the means and covariances of the two classes, develop an MAP classifier and test it on the data generated in step 5 of exercise 1. Report the classification accuracy.

```
The accuracy of the ML classifier is:  73.0 %
The accuracy of the MAP classifier is:  82.0 %
```

The accuracy of the MAP Classifier was higher than the one for the ML classifier by 9% (went up from 73% to 82%)

3. (5 pts) Based on the results, do you think that assuming the probability distributions of the two classes as Gaussian was correct? What if you remove the extra step of adding noise in your training data (step 1 exercise 1)? Do you think that would make these classifiers better? Explain.

```
The accuracy of the ML classifier (for training data without noise) is:  66.0 %
The accuracy of the MAP classifier (for training data without noise) is:  73.0 %
```

No, it was not correct since we introduced/ added random noise to the samples that have a normal distribution. However, it is reasonable to assume that the probability distribution was gaussian considering the central limit theorem, which states that the distribution of the sum/ mean of large and independent variables will approximate a normal Gaussian distribution regardless of the population's distribution.

Adding random noise to the training data enhanced the accuracy of both the ML and MAP models. This happened since introducing noise to the training dataset can help to prevent overfitting of the model, and to reduce poor generalization to new data. When noise is introduced, the model can become more robust and differentiate patterns from noise.

Another advantage of introducing noise is that it can help the training data to be more diversed, making the models less biased.

4. (5 pts) Compare the ML, MAP, MMD, and MED classifiers based on the classification accuracy. Which classifier is the best? Could the inferior classifiers be better for different data? Explain.

ML Classifier: 73%
MAP Classifier: 82%
MMD Classifier: 81%
MED Classifier: 76%

These results were obtained from the training dataset with 100 samples that had noise. Based on the classification accuracy, the MAP and MMD Classifier led to better results with a highest accuracy percentage, followed by MED Classifier and the ML classifier. This result makes sense since the MMD classifier uses the minimum Mahalanobis distance between data points and class centroids, being a useful classifier when classes have different covariances and for non-Gaussian data.In contrast to the Maximum Likelihood (ML) classifier, which assumes that all classes are equally likely, the MAP classifier uses prior probabilities to determine the class distribution in the training data. This can potentially improve accuracy, especially in situations where classes are imbalanced.

It can be assumed that the MAP, ML and MED classifiers were more sensitive to noise, leading to overfiting the noise in the training data. Both the ML and MAP Classifiers are expected to perform better when working with data that has a Gaussian distribution. Having a correct knowledge of the class distribution will impact the accuracy of these models. On the other hand, the MED classifier will perform better for data that do not have strong covariance differences, and that has no noise.

Assignment 2 - Q3

October 17, 2023 5:27 PM

In this exercise, you will be estimating the parameters of different distributions using the maximum likelihood principle. You can assume 1-d data, which means x is not a vector but just a real number.

1. (10 pts) Let's assume that the probability distribution of your dataset is an exponential distribution (Wiki). In this case, you only need to estimate a single parameter λ. Use MLE to derive an expression for finding λ.

Probability density function (PDF) of an exponential distribution:

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

where λ is the parameter of distribution

Knowing that a maximum likelihood estimator $\hat{\theta}$ of θ_0 is obtained as a solution of a maximization problem

$$\Rightarrow \quad \hat{\theta} = \arg\max_{\theta} \underbrace{p(\{x_i\}|\theta)}_{\text{likelihood function}}$$

$\Rightarrow$ for λ (rate parameter):

$$\hat{\lambda} = \arg\max_{\lambda} L(\lambda; x_1, \dots, x_n)$$

$$\Rightarrow \quad \frac{\partial}{\partial \lambda} L(\lambda; x_1, \dots, x_n) = 0 \quad \text{To find maximum}$$

First, we need to find the log likelihood function:

$$L(\lambda; x_1, \dots, x_n)$$

This can be obtained by taking the natural logarithm of the likelihood function

$$\begin{aligned} L(\lambda; x_1, \dots, x_n) &= \ln(L(\lambda; x_1, \dots, x_n)) \\ &= \ln\left(\lambda^n \exp\left(-\lambda \sum_{i=1}^{n} x_i\right)\right) \\ &= \ln(\lambda^n) + \ln\left(\exp\left(-\lambda \sum_{i=1}^{n} x_i\right)\right) \\ &= n \ln \lambda - \lambda \sum_{i=1}^{n} x_i \end{aligned}$$

$$\Rightarrow \quad \frac{\partial}{\partial \lambda} L(\lambda; x_1, \dots, x_n) = \frac{\partial}{\partial \lambda}\left(n \ln(\lambda) - \lambda \sum_{i=1}^{n} x_i\right)$$

$$\Rightarrow \quad \frac{\partial}{\partial \lambda}\left(n \ln \lambda - \lambda \sum_{i=1}^{n} x_i\right) = \frac{\partial}{\partial \lambda} n \ln(\lambda) - \frac{\partial}{\partial \lambda} \lambda \sum_{i=1}^{n} x_i$$

$$\Rightarrow \quad n \frac{1}{\lambda} - \sum_{i=1}^{n} x_i$$

$$\Rightarrow \quad \frac{n}{\lambda} - \sum_{i=1}^{n} x_i = 0$$

$$\frac{n}{\lambda} = \sum_{i=1}^{n} x_i$$

$$\boxed{\hat{\lambda}_n = \frac{n}{\sum_{i=1}^{n} x_i}} \Rightarrow \text{inverse mean}$$

2. (10 pts) Now, let's assume the probability distribution of your dataset is uniform (Zhang Ch. 19.8). In this case, you will estimate two parameters a and b. Use MLE to derive an expression for finding the two parameters.

For a random variable $X \Rightarrow X \sim U[a, b]$

Assume uniform probability distribution

$$p(x) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq x \leq b \\ 0 & \text{for } x < a \text{ or } x > b \end{cases}$$

Suppose we are given random samples $x_1, x_2, \dots, x_n$ drawn from $p(x)$.
Being the unknown parameters the two endpoints

$$\theta = \begin{bmatrix} a \\ b \end{bmatrix}$$

Such that, the maximum likelihood estimate for a and b is given by:

$$\theta_{ML} = \arg\max_{\theta} p(\{x_i\}|\theta) = \arg\max_{a,b} \prod_{i=1}^{n} p(x_i|a,b)$$

Taking the logarithm

$$\Rightarrow \quad \frac{\partial}{\partial \theta} \log\left\{\prod_{i=1}^{n} p(x_i|\theta)\right\}$$

$$\Rightarrow \quad \frac{\partial}{\partial \theta} \log\left\{\prod_{i=1}^{N} \frac{1}{b-a}\right\} \quad \text{where } \prod_{i=1}^{N} \frac{1}{b-a} = \frac{1}{(b-a)^n}$$

$$\Rightarrow \quad \frac{\partial}{\partial \theta} \log \frac{1}{(b-a)^n}$$

$$\Rightarrow \quad \frac{\partial}{\partial \theta} \log (b-a)^{-n}$$

$$\Rightarrow \quad \frac{\partial}{\partial \theta}\left[-n \log(b-a)\right]$$

Since θ is a function of a and b,

$$\Rightarrow \frac{\partial f(\theta)}{\partial \theta} = \begin{bmatrix} \frac{\partial f(\theta)}{\partial a} & \frac{\partial f(\theta)}{\partial b} \end{bmatrix}^T$$

where

$$\frac{\partial f(\theta)}{\partial a} = \frac{\partial}{\partial a}\left(-n \log(b-a)\right)$$

$$\frac{\partial f(\theta)}{\partial a} = -n \frac{1}{(b-a)}(-1)$$

$$\frac{\partial f(\theta)}{\partial a} = \frac{n}{b-a}$$

And,

$$\frac{\partial f(\theta)}{\partial b} = \frac{\partial}{\partial b}\left(-n \log(b-a)\right)$$

$$\frac{\partial f(\theta)}{\partial b} = -\frac{n}{b-a}$$

We know $\frac{\partial f(\theta)}{\partial \theta} = 0$

$$\Rightarrow \quad \frac{n}{b-a} - \frac{n}{b-a} = 0 \quad \Rightarrow \quad \text{No information}$$

Since ML maximizes the function, we can log that:
$p(x)$ is maximized by maximizing $(b-a)$

Therefore,

$$\hat{a}_{ML} = \min\{x_i\} \qquad \hat{b}_{ML} = \max\{x_i\}$$

$$\Rightarrow \quad \boxed{\max\{p(x_i|a,b)\} = \frac{1}{b-a} = \frac{1}{\max\{x_i\} - \min\{x_i\}}}$$

3. (15 pts) We derived the expressions for the mean and standard deviation of the Gaussian distribution in class. Now, let's assume the probability distribution of your dataset is not just a single Gaussian distribution but a product of a Gaussian distribution and an exponential distribution:

$$p(X|\theta) = \frac{1}{2}[\mathcal{N}(X|\theta_1, \theta_2) \times f(x; \lambda)] \tag{2}$$

In this case, you will be estimating three parameters (θ_1, θ_2, and λ), where θ_1 and θ_2 are the mean and variance of the Gaussian distribution. Using MLE, derive an expression for finding the three parameters of your probability distribution defined by eq. (2).

$$p(x|\lambda) = \frac{1}{2}\left[\mathcal{N}(x|\underbrace{\theta_1, \theta_2}_{\mu, \sigma^2}) \cdot f(x;\lambda)\right]$$

where $\theta = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T$

for the dist we say $p(x|\lambda) = \frac{1}{2}\left[\frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \cdot \lambda e^{-\lambda x}\right], x \geq 0$

where μ - mean, σ - STD, λ - parameter of exponential component

The likelihood generated from this distribution is given by:

$$L(\mu, \sigma, \lambda) = \prod_{i=1}^{n} f(x_i; \mu, \sigma, \lambda)$$

$$\boxed{L(\mu, \sigma, \lambda) = \prod_{i=1}^{n}\left(\frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2} \cdot \lambda e^{-\lambda x_i}\right)}$$

To find the MLE estimates for the parameters μ, σ and λ, we need to maximize the log-likelihood function:

$$l(\mu, \sigma, \lambda) = \sum_{i=1}^{n}\left[\log 1 - \frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2 - \frac{1}{2}\log\sqrt{2\pi}\,\sigma + \log(\lambda) - \lambda x_i\right]$$

$$\boxed{l(\mu, \sigma, \lambda) = \sum_{i=1}^{n}\left[\log 1 - \frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2 - \frac{1}{2}\log(2\pi) - \log(\sigma) + \log(\lambda) - \lambda x_i\right]}$$

Expression to find these parameters

Differentiating l with respect to μ, σ, λ:

- μ:

$$\frac{\partial l}{\partial \mu} = \sum_{i=1}^{n} \frac{x_i - \mu}{\sigma^2}$$

$$\frac{\partial l}{\partial \mu} = \frac{1}{\sigma^2} \sum_{i=1}^{N} (x_i - \mu) = 0$$

$$\sum_{i=1}^{n} (x_i) - n\mu = 0$$

$$\Rightarrow \quad \hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$$

- σ:

$$\frac{\partial l}{\partial \sigma} = \frac{1}{2} \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{\sigma^2} - \frac{n}{\sigma^2}$$

Setting $\frac{\partial l}{\partial \sigma} = 0$ and plugin in $\hat{\mu}_n$:

$$\frac{\partial l}{\partial \sigma} = \frac{1}{2} \sum_{i=1}^{n} \frac{(x_i - \hat{\mu}_n)^2}{\sigma^2} - \frac{n}{\sigma^2} = 0$$

$$\sum_{i=1}^{n} \frac{(x_i - \hat{\mu}_n)^2}{\sigma^2} = \frac{n}{\sigma^2}$$

$$\sum_{i=1}^{n} (x_i - \hat{\mu}_n)^2 = n\sigma^2$$

$$\hat{\sigma}_{ML} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu}_n)^2$$

- λ:

From Q1, we know:

$$\hat{\lambda} = \frac{n}{\sum_{i=1}^{n} x_i}$$

4. (10 pts) Generate 100 samples for two classes using the following statistics:

- Class 1: $N_1 = 5$, $\mu_1 = 0.5$, $\sigma_1 = 1$
- Class 2: $N_2 = 5$, $\mu_2 = 5$, $\sigma_2 = 3$

Using the formulae derived for the above three distributions, develop three ML classifiers using the generated data. Now, generate 50 samples for each class, and add white noise to each of the samples by using *numpy.random.normal* with 0 mean and 1 standard deviation. Test the three ML classifiers on the noisy test data. Which of the three ML classifiers is the best? Why?

```
Accuracy for ML classifier with an exponential distribution: 0.8 %
Accuracy for ML classifier with an uniform distribution: 50.0 %
Accuracy for ML classifier that combines a Gaussian and an exponential distribution: 84.0 %
```

The ML classifier with a Gaussian and exponential distribution led to the highest accuracy (84%). This result make sense since this classifier considers both Gaussian and exponential distributions. Therefore, this will be good for data exhibiting a similar behaviour. In contrast, the uniform distribution classifier is simple and would not capture the variation of data observed in real-world datasets, leading to poor accuracies. Finally, the exponential classifier will be better for data that has an exponential distribution. For this question, the samples were generated using a normal distribution.