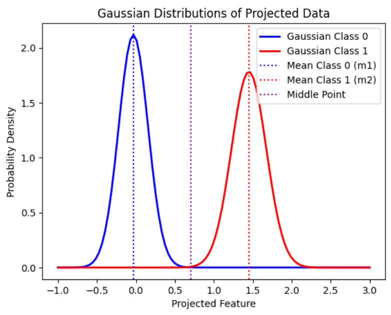
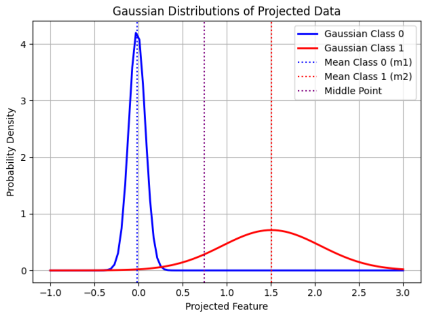
黃偉祥 X1136010

Possible reason why the accuracy or F1-score change between Perceptron and LDA.

- Because the Perceptron is not guaranteed to reduce the total error monotonically, while LDA use discriminant vector (w) to get the optimal solution.
- Meaning that LDA can get a best line to separate the data using some mathematics method in limited time and epoch, but Perceptron may need to try many epochs until converges.
- And in the Lab3 I tried {1,3,5,9,11} epochs with Perceptron algorithm, the best prediction is 5 and not 11 epochs. It may have better result if keep increasing the iteration of training until convergence. Even if 5 epochs have 0.68 f1-score but LAD only train once and get 0.69 f1-score. (On Kaggle)

Does MAP help? Why?

- In this dataset, no, the simplest way is to check their f1-score, since both of them get the same 0.69 f1-score on Kaggle.
- LDA and LDA with MAP use the same W to project the data, the different of LDA and LDA with MAP is the boundary.
- Since we already project to the line that S_w (within-class variance) is minimum, and the maximum distance of m_1 and m_2, so using the middle point of m_1 and m_2 to be the boundary is the simplest and good enough way to separate the data like Graph_A, if the dataset is uniform distribution.
- But, LDA with MAP may perform better in others dataset, like Graph_B.

	
Graph_A	Graph_B

Summarize how you solve the difficulty and your reflections

- First is in the Perceptron.predict, I use return 1 if self.linear_combination(X) >= 0 else 0, but this is only for 1 data, not for an array of data. So I use np.where() to solve the problem.
- When calculating fisher_discriminant(X,y), it really make me confusing how to calculate the S_w, from the equation it looks like after calculation should be a number(summation) and not a matrix. So, I tried many different approaches and finally thanks to ChatGPT, I realize the result of S_w should be a matrix and using np.sum() is not the way, it should calculate through matrix and resulting a matrix too.