Problem 1 (Create your own Perceptron algorithm)

(1) Use Python to generate a 2D ($\boldsymbol{x}_i \in \mathcal{R}^2$) linearly separable data set with 50 data points. The data set should have approximately half positive and half negative instances. Create a scatter plot to visualize the data set. Split the data set into training set (60%) and test set (40%).

(2) Use Python (not sklearn package) to create the **Batch** Perceptron training algorithm and use the training data set in a.(1) to train a Perceptron model. Plot the error function curve when the training process converges. Create a plot that shows the training instances and the learnt decision boundary.

(3) Use the test set in a.(1) to test the trained model in a.(2) and calculate the accuracy (error rate) of the trained model.

(4) Use Python (not sklearn package) to create the **sequential** Perceptron training algorithm and use the training data set in a.(1) to train a Perceptron model. Plot the weights vs iterations curve when the training process converges. Create a plot that shows the training instances and the resulting decision boundary.

(5) Use the test set in a.(1) to test the trained model in a. (4) and calculate the accuracy (error rate) of the trained model.

(6) Show how you select learning rate during the training process of a.(2) or a.(4) and demonstrate how the choice of the learning rate is affecting the convergence of the training process.

Problem 2 (Use Perceptron model in sklearn to do classification)

Part a:

Use the *make_classification* method in *sklearn* to create a ***binary linearly separable*** data set with two (2) features and 100 data points. Visualize the data set.

Split the dataset into to training and test data sets using the *train_test_split* method in *sklearn*.

Use the training set to train the *Perceptron* model in *sklearn* then use the trained model to make prediction on the test set.

Calculate the ***accuracy*** of the classification

Visualize the ***decision boundary*** of the trained *Perceptron* model on top of the test data points.

Part b:

Use the *make_classification* method in *sklearn* to create a ***binary non linearly separable*** data set with two (2) features and 100 data points. The data set should have ***class imbalance***. Visualize the data set.

Split the dataset into to training and test data sets using the *train_test_split* method in sklearn.

Use the training set to train the *Perceptron* model in sklearn then use the trained model to make prediction on the test set.

Use *sklearn.metrics* to show the following on the classification results:

Accuracy, Precision, Recall, F1 score

The *classification_report* and the *confusion_matrix*

Visualize the ***decision boundary*** of the trained Perceptron model on top of the test data points.