Task 1 – Dataset (Dry Beans)

- The data set consists of 50 samples from each of three categories of Dry beans (Bomay, Cali and Sira).

- Five features were measured from each sample: Are, Perimeter, MajorAxisLength, MinorAxisLength and roundness *(in millimeter)*.

Bomay

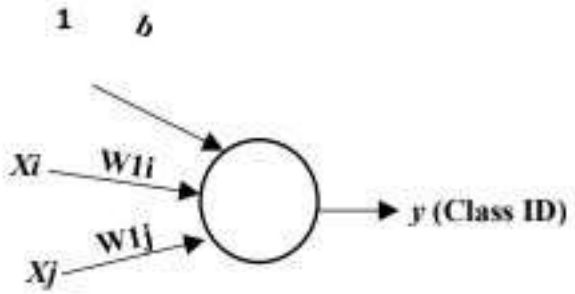
Cali

Sira

Task 1 - GUI

1. **User Input:**
   - Select two features
   - Select two classes (C1 & C2 or C1 & C3 or C2 & C3)
   - Enter learning rate (eta)
   - Enter number of epochs (m)
   - Enter MSE threshold (mse_threshold)
   - Add bias or not (Checkbox)
   - Choose the used algorithm perceptron or Adaline (radio button)

2. **Initialization:**
   - Number of features = 2.
   - Number of classes = 2.
   - Weights + Bias = small random numbers



3. **Classification:**
   - Sample (single sample to be classified).

Task 1 - Description

1. Implement the Perceptron learning algorithm

- Single layer neural networks which can be able to classify a stream of input data to one of a set of predefined classes.

- Use the Dry Beans data in both your training and testing processes. (Each class has 50 samples: train NN with 30 non-repeated samples randomly selected, and test it with the remaining 20 samples)

Task 1 - Description

2. Implement the Adaline learning algorithm using MSE

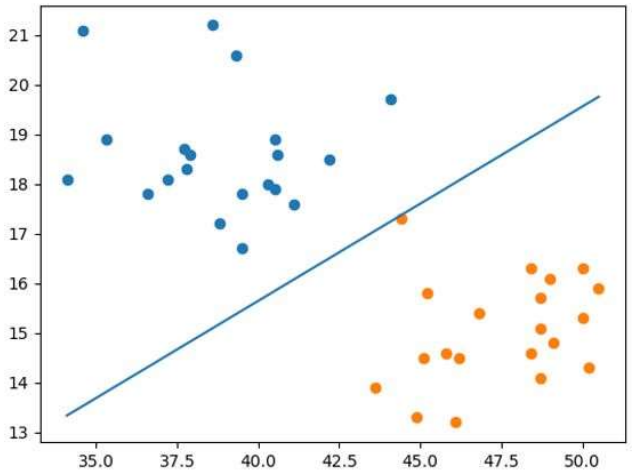
- Single layer neural networks which can be able to classify a stream of input data to one of a set of predefined classes.

- Use the Dry beans data in both your training and testing processes. (Each class has 50 samples: train NN with 30 non-repeated samples randomly selected, and test it with the remaining 20 samples)

Task 1 - Description

3. After training

- Draw a line that can discriminate between the two learned classes. You should also scatter the points of both classes to visualize the behavior of the line.



- Test the classifier with the remaining 20 samples of each selected classes and find confusion matrix and compute overall accuracy.

Task 1 – Workflow (Perceptron)

- **Training Phase: (repeat the following *m* epochs)**

  Assuming that we have ***n*** training samples $\{sample_i : i = 1 \rightarrow \boldsymbol{n}\}$

  - Fetch features (***x***) of $sample_i$, and its desired output (***d***)
  - Calculate the net value (***v***),
  - Calculate actual output (***y***) using ***signum*** activation function,
  - Calculate the ***error*** $= \boldsymbol{d} - \boldsymbol{y}$,
  - Update the weights (***new weights = old weights + eta * error * x***), note: old weights is $\begin{bmatrix} b \\ W1i \\ W1j \end{bmatrix}$

- **Draw line:** line equation is ***W1i * Xi + W1j * Xj + b = 0***

Task 1 – Workflow (Adaline)

- **Training Phase: (repeat the following *m* epochs)**

  Assuming that we have ***n*** training samples $\{sample_i: i = 1 \rightarrow n\}$

  - Fetch features (***x***) of $sample_i$, and its desired output (***d***)
  - Calculate the net value (***v***),
  - Calculate actual output (***y***) using ***Linear*** activation function,
  - Calculate the ***error*** = ***d*** – ***y***,
  - Update the weights (***new weights*** = ***old weights*** + ***eta*** * ***error*** * ***x***), note: old weights is $\begin{bmatrix} b \\ W1i \\ W1j \end{bmatrix}$

- **Draw line:** line equation is ***W1i*** * ***Xi*** + ***W1j*** * ***Xj*** + ***b*** = ***0***

Task 1 - Workflow

- **Testing Phase:**
  1. Given a sample x
  2. Calculate the net value (v),
  3. Calculate actual output (y) using ***signum*** activation function,
  4. **Output:** y (Class ID).

- **Evaluation:** build the confusion matrix and overall accuracy

Task 1 - Notes

1. You will be asked to deliver a .rar folder containing all your code files (.py), dataset and the visualization and analysis report.

2. In the report you should have screenshots/plots of the visualizations and a written analysis of what you understood from each of these visualizations and how the features discriminate or not between classes. You should have a plot and analysis for each combination (5 combinations for each algorithm)

3. At the end of the report, you should mention which features achieved the highest accuracy after running your algorithm.

4. **You should not drop any row from the dataset.**

5. Using scikit-learn metrics library or any similar built-in function for the confusion matrix is not allowed.

Task 1 - Notes

1. Lab 3 including Task 1 description will be available **Thursday 26/10/2023**.
   Task Deadline: 7/11/2023 11:59 PM

2. Please try to write well designed code.

3. Separate the logic (generation, loading, and classification) from the UI.

4. Writing well-documented, readable, maintainable, and extensible code is an extremely important skill you must master before graduating. So, don't be lazy!

5. Cheating will not be tolerated.