To implement the Perceptron Learning algorithm, Pocket algorithm, to solve the Binary classification problem.

PROJECT DESCRIPTION

- Programming language used : Python
- Data Structure : Lists, matrix
- File Name : PLA.py, PocketAlgo.py, LinearRegression.py, LogisticRegression.py
- Inputs: classification.txt, linear-regression.txt
- Output:
    - Perceptron Learning Algorithm – Weight Matrix such that all the points in the input set are linearly separable.
    - Pocket Algorithm – Weight Matrix such that all the points in the input set are not linearly separable.

IMPLEMENTATION

- Modules created :

A. Percptron Learning Algorithm:

1. ReadFile(): to save the input file into List of points for the given dimensions in the global variable Data_List and output_coordinate_list for the classification label.
2. constraintVerify(X,Y) : for each datapoint X checks if w*X is positive and as per the classified constraint Y.
3. updateW(i): For every i^{th} data point whose value for w*X and Y do not map, the weights are updated with the formula : w = w+ alpha * X (for negative instances) and w=w-alpha*X(for positive misclassified instances).
4. Perceptron(w_old): Calling function to check for each data point and adjust their weights until all are satisfied and there are no misclassified points further.

B. Pocket Algorithm:

1. ReadFile(): to save the input file into List of points for the given dimensions in the global variable Data_List and output_coordinate_list for the classification label.
2. constraintVerify(X,Y) : for each datapoint X checks if w*X is positive and as per the classified constraint Y.
3. updateW(i): For every i^{th} data point whose value for w*X and Y do not map, the weights are updated with the formula : w = w+ alpha * X (for negative instances) and w=w-alpha*X(for positive misclassified instances).
4. Perceptron(w_old): Calling function to check for each data point and adjust their weights until all are satisfied. The program converges at 7000 iterations as there are points that are always misclassified. The method gives the least number of misclassified constraints during the 7000 iterations and the corresponding weight.

5. Weightupdate(m): to store the weights after every iteration in a global variable weightMatrix.
6. plotting(): To plot the graph of iteration vs misclassified points.

- Termination Condition :
  - PLA : Recursive program call to Perceptron, terminates when all the points are correctly classified as per the updated weight.
  - PocketAlgo: Terminates when 7000 iterations are complete or all the points in the input are correctly classified.

- Complexity:
  - The PLA weight calculation is done in O(n log n) time, where n is the size of the input data set.
  - The Pocket Algorithm weight calculation is done in $O(n^2)$ time = O (7000 iterations*2000data points)
  - The Logistic Regression Algorithm weight calculation is done in $O(n^2)$ time = O (7000 iterations*2000data points)
  - The Linear Regression weight calculation is done in O(1) time.

- Result Interpretation:
  - The PLA output presents the weight vector which satisfies the binary classification of points into positive and negative classes.
  - The PocketAlgo output gives the weight for the least number of unclassified points as the points cannot be classified further.