## **Perceptron: Toy Example**

Perceptron Learning Algorithm in action

1. Dataset

|  |  |  |
| --- | --- | --- |
| x1 | x2 | y |
| -1 | -1 | 0 |
| -5 | -2.5 | 0 |
| -7.5 | 7.5 | 0 |
| 10 | 7.5 | 1 |
| -2.5 | 12.5 | 0 |
| 5 | 10 | 1 |
| 5 | 5 | 1 |

1. P ⇒ Green points, N ⇒ Red points
2. Decision boundary line is given by w1x1 + w2x2 - b >= 0
   1. Or x2 = -(w1/w2)x1 + (b/w2)
   2. Can be rewritten as x2 = mx + c
   3. Where m = -(w1/w2) and c = (b/w2)
3. Initialize w randomly
   1. w1 = 1.00, w2 = 1.00, b = 5.00 ⇒ m = -1.00, c = 5.00
4. The line is x2 = -x1 + 5
5. While !convergence do:
   1. Pick random x ∈ P ∪ N
   2. If x ∈ P and ni=0 wixi < 0 then, w = w + x; **end**
   3. If x ∈ N and ni=0 wixi >= 0 then, w = w - x; **end**
   4. Consider x = [x0;x1;x2] and w = [w0;w1;w2], where x0 bias term is always 1
   5. On the 5th training example, condition c isn’t satisfied, so we recalculate w = w - x
   6. w1 = 3.5, w2 = -11.5, b = 4.00 ⇒ m = 0.3, c = -0.35
   7. The line changes, causing a new error on the 6th training example, so we calculate w = w+x
   8. w1 = 8.5, w2 = -1.5, b = 5.00 ⇒ m = 5.67, c = -3.33
   9. The resulting line predicts all examples perfectly, thus convergence is reached