

```

1  while True: #Покоординатный спуск
2      iDelta = 0
3      while iDelta < ordinateCount:
4          ww = calcCutoffDistance(classesCount, instancesMax, ordinateCount,
5                                  Target, Opposite, vectorWeightsCurr, argClasses)
6          countCutoffPrev = ww[0]
7          distanceCutoffPrev = ww[1]
8          deltaMultiPrev = 0
9          weightsOld = vectorWeightsCurr[iDelta] #Проверка направления спуска
10         deltaMultiCurr = calcDescentDirection(classesCount, instancesMax,
11                                                 ordinateCount, Target, Opposite, vectorWeightsCurr, argClasses)
12         while True:
13             deltaMultiCurr = deltaMultiCurr * 2
14             vectorWeightsCurr[iDelta] = weightsOld + deltaMultiCurr + deltaMultiPrev
15             ww = calcCutoffDistance(classesCount, instancesMax, ordinateCount, Target,
16                                     Opposite, vectorWeightsCurr, argClasses)
17             countCutoffCurr = ww[0]
18             distanceCutoffCurr = ww[1]
19             if countCutoffCurr > countCutoffPrev:
20                 countCutoffPrev = countCutoffCurr
21                 distanceCutoffPrev = distanceCutoffCurr
22                 deltaMultiPrev += deltaMultiCurr
23                 deltaMultiCurr = calcDescentDirection(classesCount, instancesMax,
24                                                         ordinateCount, Target, Opposite, vectorWeightsCurr, argClasses)
25             elif countCutoffCurr < countCutoffPrev:
26                 deltaCutoffDistance[0][iDelta] = vectorWeightsCurr[iDelta] -
27                 deltaMultiCurr - weightsOld
28                 deltaCutoffDistance[1][iDelta] = countCutoffPrev
29                 vectorWeightsCurr[iDelta] = weightsOld
30                 break
31             elif distanceCutoffCurr > distanceCutoffPrev:
32                 deltaCutoffDistance[0][iDelta] = vectorWeightsCurr[iDelta] -
33                 deltaMultiCurr - weightsOld
34                 deltaCutoffDistance[1][iDelta] = countCutoffPrev
35                 vectorWeightsCurr[iDelta] = weightsOld
36                 break
37             else:
38                 countCutoffPrev = countCutoffCurr
39                 distanceCutoffPrev = distanceCutoffCurr
40             iDelta += 1
41         iDelta = 1
42         maxCutoff = deltaCutoffDistance[1][0]
43         maxCutoffIndex = 0
44         while iDelta < ordinateCount:
45             if maxCutoff < deltaCutoffDistance[1][iDelta]:
46                 maxCutoff = deltaCutoffDistance[1][iDelta]
47                 maxCutoffIndex = iDelta
48             iDelta += 1
49         iDelta = 0
50         condCycle = 0
51         minCutoffDistance = abs(deltaCutoffDistance[0][maxCutoffIndex])
52         maxCutoffIndex = ordinateCount
53         while iDelta < ordinateCount:
54             condCycle += abs(deltaCutoffDistance[0][iDelta])
55             if maxCutoff == deltaCutoffDistance[1][iDelta]:
56                 if minCutoffDistance >= abs(deltaCutoffDistance[0][iDelta]):
57                     maxCutoffIndex = iDelta
58             iDelta += 1
59         vectorWeightsCurr[maxCutoffIndex] += deltaCutoffDistance[0][maxCutoffIndex]
60         if condCycle == 0:
61             break
62     contrastingWeights(classesCount, instancesMax, ordinateCount, Target, Opposite,
63                         vectorWeightsCurr, argClasses)
64     valueDoorstep = calcBiasDoorstep(classesCount, instancesMax, ordinateCount, Target,
65                                       Opposite, vectorWeightsCurr, argClasses)
66     print(vectorWeightsCurr)

```