

A

Pseudocode for the SMO Algorithm

```
target = desired output vector
point = training point matrix

procedure takeStep(i1,i2)
  if (i1 == i2) return 0
  alph1 = Lagrange multiplier for i1
  y1 = target[i1]
  E1 = SVM output on point[i1] - y1 (check in error cache)
  s = y1*y2
  Compute L, H
  if (L == H)
    return 0
  k11 = kernel(point[i1],point[i1])
  k12 = kernel(point[i1],point[i2])
  k22 = kernel(point[i2],point[i2])
  eta = 2*k12-k11-k22
  if (eta < 0)
  {
    a2 = alph2 - y2*(E1-E2)/eta
    if (a2 < L) a2 = L
    else if (a2 > H) a2 = H
  }
  else
  {
    Lobj = objective function at a2=L
    Hobj = objective function at a2=H
    if (Lobj > Hobj+eps)
      a2 = L
    else if (Lobj < Hobj-eps)
      a2 = H
    else
      a2 = alph2
  }
  if (|a2-alph2| < eps*(a2+alph2+eps))
```

```

    return 0
    a1 = alph1+s*(alph2-a2)
    Update threshold to reflect change in Lagrange multipliers
    Update weight vector to reflect change in a1 & a2, if linear SVM
    Update error cache using new Lagrange multipliers
    Store a1 in the alpha array
    Store a2 in the alpha array
    return 1
endprocedure

procedure examineExample(i2)
    y2 = target[i2]
    alph2 = Lagrange multiplier for i2
    E2 = SVM output on point[i2] - y2 (check in error cache)
    r2 = E2*y2
    if ((r2 < -tol && alph2 < C) || (r2 > tol && alph2 > 0))
    {
        if (number of non-zero & non-C alpha > 1)
        {
            i1 = result of second choice heuristic
            if takeStep(i1,i2)
                return 1
        }
        loop over non-zero and non-C alpha, starting at random point
        {
            i1 = identity of current alpha
            if takeStep(i1,i2)
                return 1
        }
        loop over all possible i1, starting at a random point
        {
            i1 = loop variable
            if takeStep(i1,i2)
                return 1
        }
    }
    return 0
endprocedure

main routine:
    initialize alpha array to all zero
    initialize threshold to zero
    numChanged = 0
    examineAll = 1
    while (numChanged > 0 | examineAll)
    {

```

```
numChanged = 0
if (examineAll)
    loop I over all training examples
        numChanged += examineExample(I)
else
    loop I over examples where alpha is not 0 & not C
        numChanged += examineExample(I)
if (examineAll == 1)
    examineAll = 0
else if (numChanged == 0)
    examineAll = 1
}
```

©1998, John Platt, reprinted with permission