SVM Pseudo-Code

Lindsay Forestell
January 27, 2019

Inputs

C := Regularization Parameter

tol := Tolerance for accepting/rejecting α_i

max_passes := Number of times to try to iterate over α_i before halting because it's not changing.

(x_i,y_i) := Training Data, where y has to be ±1

Initialize

$\alpha_i = 0$

b = 0

passes = 0

Kernel

$$K_{i,j} \equiv \langle \phi(x_i), \phi(x_j) \rangle$$

For linear SVM:

$$K_{i,j} \equiv \langle x_i, x_j \rangle = x_i^T x_j$$

Other Kernels:

Polynomial:

$$K_{i,j} = (1 + \langle x_i, x_j \rangle)^d$$

Radial:

$$K_{i,j} = exp(-\gamma(x_i - x_j)^2)$$

Classifying

$$f(x_{test}) = \sum_{m=1}^{n}[\alpha_m y_m K_{test,m}] + b$$

$$y_{test} = sign(f(x_{test}))$$

Update Algorithm

```
**WHILE** passes < max_passes:
    n_changed_alpha = 0
    **FOR** i=1,...,n:
        $E_i = \sum_{m=1}^{n}[\alpha_m y_m K_{i,m}] + b - y_i$
        **IF** ($y_i E_i <$ -tol AND $\alpha_i$ <C) OR ($y_i E_i >$ tol AND $\alpha_i > 0$):
            select j≠i randomly from the rest
            $E_j = \sum_{m=1}^{n}[\alpha_m y_m K_{j,m} + b] - y_j$
            $\alpha_j^{old} = \alpha_j$
            $\alpha_i^{old} = \alpha_i$
            **IF** $y_i \neq y_j$:
                L = max(0,$\alpha_i - \alpha_j$)
                H = min(C, C - ($\alpha_i - \alpha_j$))
            **ELSE**:
                L = max(0,$\alpha_i + \alpha_j$ - C)
                H = min(C, $\alpha_i + \alpha_j$)
            **IF** L = H: **continue** to next i
```

```
      $\eta = 2K_{i,j} - K_{i,i} - K_{j,j}$
      **IF** $\eta \geq 0$: **continue** to next i
      $\alpha_j = \alpha_j^{old} - \frac{y_j(E_i - E_j)}{\eta}$
      **IF** $\alpha_j > H$: $alpha_j = H$
      **ELSE IF** $L \leq \alpha_j \leq H$: $\alpha_j = \alpha_j$
      **ELSE IF** $\alpha_j < L$: $\alpha_j = L$
      **IF** $|\alpha_j - \alpha_j^{old}| < 10^{-5}$: **continue** to next i
      $\alpha_i = \alpha_i^{old} + y_i y_j(\alpha_j^{old} - \alpha_j)$
      $b_i = b - E_i - y_i(\alpha_i - \alpha_i^{old})K_{i,i} - y_j(\alpha_j - \alpha_j^{old})K_{i,j}$
      $b_j = b - E_j - y_j(\alpha_j - \alpha_j^{old})K_{j,j} - y_i(\alpha_i - \alpha_i^{old})K_{i,j}$
      **IF** $0 < \alpha_i < C$: $b = b_i$
      **ELSE IF** $0 < \alpha_j < C$: $b = b_j$
      **ELSE**: $b = \frac{b_i + b_j}{2}$
      n_changed_alpha += 1
    end **IF**
  end **FOR**
  **IF** n_changed_alpha = 0: passes += 1 **ELSE**: passes = 0
end **WHILE**
```