Multi class perceptron

In multi class perceptron our goal is to make a weight matrix W and then for each example we predict $r = argmax_r(Wx)_r$. In each update we use the update rule
$W = W + U^t$ where
$U^t_{r,j} = x_{rj}(1[y == r] - 1[y \hat{==} r)$

Hinge Loss and mistake bound

Hinge loss

Hinge loss is defined by
$l(W;(x,y)) = max_{r \in k-y}[1-(Wx)_y + (Wx)_r]_+$
Here we can define Loss of perceptron as
$L = \sum_1^T l(W;(x,y))$

Complexity

complexity of a weight matrix is defined as 2 times the square of its L-2 norm
$D = 2\sum\sum W(i,j)^2$

Mistake bound

The mistake bound for multiclass perceptron is
$M \leq L + D + \sqrt{LD}$

Banditron

In our problem , we do not know what is the correct label in case our prediction is wrong so we cannot apply multi class perceptron . In this case similar to exp3 , we add a exploration factor. So first we construct a extimator for matrix U

$$\tilde{U_{rj}} = x_j(\frac{1(y == r)1(\tilde{y} == r)}{P(r)} - 1(y \hat{==} r))$$

Here $\tilde{y}$ is our prediction obtained from the probability vector $P(r) = (1 - \gamma)1(r == \hat{y} + \gamma/k$, here γ is our exploration constant.

Banditron algorithm

for t = 1,2,...T:

$\hat{y} = argmax_r(W^t x)_r$

make the probability array $P(r) = (1 - \gamma)1(r == \hat{y} + \gamma/k$

sample $\tilde{y}$ according to P

make the update matrix

$$\tilde{U}_{rj} = x_j(\frac{1(y == r)1(\tilde{y} == r)}{P(r)} - 1(y \,\hat{==}\, r))$$

$W = W + \tilde{U}$

Theorem 1

If our dataset satisfies $||x|| \leq 1$, then the mistake bound of banditron will be and there are k classes,

$$E[M] \leq L + \gamma T + 3 max kD/\gamma, \sqrt{D\gamma T} + \sqrt{\frac{kDL}{\gamma}}$$

Lemma 5

$$E[||U||^2] \leq 2||x_t||^2(k/\gamma 1||y \neq \hat{y}|| + \gamma 1||y \neq \hat{y}||)$$

Halving algorithm

Here we take a cover of hypothesis H and at every iteration remove the hypothesis that gave wrong answer $H^{'}$. This only works for separable case as otherwise all the hypothesis will be removed

Theorem 7

There exists a deterministic algorithm (in the bandit setting), taking D as input, which makes at most $O(k^2 d \ln(Dd))$ mistakes on any sequence (where $||x_t||1$) that is linearly separable.

Another trick to reduce the mistake bound is to remap x onto a lower dimension $d' = O(D \ln{(T+k)}/\delta)$. Here the algorithm will give a mistake bound of $O(k^2 D \ln{(T+k)}/\delta(\ln D + \ln \ln{(T+k)}/\delta))$ with probability of $1 - \delta$