Assignment 5 posted

Quiz:

1) Why is it harder to adapt the perceptron algorithm to handle the weighted classification case?

2) The bound for AVA is $2(K-1)\varepsilon$; the bound for OVA is $(K-1)\varepsilon$. Is OVA always better? Why, or why not?

Reductions

Don't solve complicated problems from zero; Find a way to change a hard problem into an easy problem and then "analyze the damage"

What are the hard problems?

- Different errors for different outcomes

- Multiple classes

(What are the error rates you are getting for perception on primary tumor?)

OVA: One versus All

Algorithm 15 ONEVERSUSALLTRAIN($\mathbf{D}^{multiclass}$, BINARYTRAIN)

```
for i = 1 to K do
    D^bin ← relabel D^multiclass so class i is positive and ¬i is negative
    f_i ← BinaryTrain(D^bin)
end for
return f_1, …, f_K
```

Algorithm 16 ONEVERSUSALLTEST($f_1, \dots, f_K, \hat{x}$)

```
score ← ⟨0, 0, …, 0⟩                  // initialize K-many scores to zero
for i = 1 to K do
    y ← f_i(x̂)
    score_i ← score_i + y
end for
return argmax_k score_k
```

Thm: If average binary error is ϵ, then OVA has error rate at most $(K-1)\epsilon$.

$f_0(x)$, $\boxed{+1}$ or -1

$f_1(x)$, $+1$ or $\boxed{-1}$

$f_2(x)$, $+1$ or $\boxed{-1}$

$f_3(x)$, $+1$ or $\boxed{-1}$

Assume $y = 0$

False negative:

$f_0(x)$, $+1$ or $\boxed{-1}$

$f_1(x)$, $+1$ or $\boxed{-1}$

$f_2(x)$, $+1$ or $\boxed{-1}$

$f_3(x)$, $+1$ or $\boxed{-1}$

$(K-1)/K$ chance of mistake

False positive: (m of them)

$f_0(x)$, $\boxed{+1}$ or -1

$f_1(x)$, $\boxed{+1}$ or -1

$f_2(x)$, $+1$ or $\boxed{-1}$

$f_3(x)$, $+1$ or $\boxed{-1}$

$\frac{m}{m+1}$ chance of mistake, but

AVA:

requires m errors!

Algorithm 17 AllVersusAllTrain($\mathbf{D}^{multiclass}$, BinaryTrain)

$f_{ij} \leftarrow \emptyset, \forall 1 \le i < j \le K$
for i = 1 **to** K-1 **do**
    $\mathbf{D}^{pos} \leftarrow$ all $\boldsymbol{x} \in \mathbf{D}^{multiclass}$ labeled i
    for j = i+1 **to** K **do**
        $\mathbf{D}^{neg} \leftarrow$ all $\boldsymbol{x} \in \mathbf{D}^{multiclass}$ labeled j
        $\mathbf{D}^{bin} \leftarrow \{(\boldsymbol{x}, +1) : \boldsymbol{x} \in \mathbf{D}^{pos}\} \cup \{(\boldsymbol{x}, -1) : \boldsymbol{x} \in \mathbf{D}^{neg}\}$
        $f_{ij} \leftarrow$ BinaryTrain($\mathbf{D}^{bin}$)
    end for
end for
return all f_{ij}s

Algorithm 18 AllVersusAllTest(all f_{ij}, $\hat{\boldsymbol{x}}$)

$score \leftarrow \langle 0, 0, \dots, 0 \rangle$ // initialize K-many scores to zero
for i = 1 **to** K-1 **do**
    for j = i+1 **to** K **do**
        $y \leftarrow f_{ij}(\hat{\boldsymbol{x}})$
        $score_i \leftarrow score_i + y$
        $score_j \leftarrow score_j - y$
    end for
end for
return $\operatorname{argmax}_k score_k$

Thm: If average binary error is ϵ, then AVA has error rate at most $2(K-1)\epsilon$.

Binary tree:

1234 vs 5678

12 vs 34   56   78

1 vs 2   3 vs 4   5 vs 6   7 vs 8

Different interpretation:

Each level predicts binary digit of label!

Error-correcting output codes

Hamming 7-4 code:

$A = d_1 \wedge d_2 \wedge d_4$

$B = d_1 \wedge d_3 \wedge d_4$

$C = d_2 \wedge d_3 \wedge d_4$

$D = d_1$

$E = d_2$

$F = d_3$

$G = d_4$

Can detect and fix any 1-bit error on 7-bit word (!!!)

Weighted classification

$$\text{Accuracy} = \sum \left[f(\vec{x}) \neq y\right] = \sum \left[f(\vec{x})=0 \wedge y=1\right] + \left[f(\vec{x})=1 \wedge y=0\right]$$

$\alpha > 1$ (weight)

$$\text{Weighted Accuracy} = \sum \alpha\left[y=1 \wedge f(\vec{x})=0\right] + \left[f(\vec{x})=1 \wedge y=0\right]$$