**Lecture 5 – White Board notes**

## **Adaboost**

### **Pictorial Representation**

|  |  |
| --- | --- |
|  | Iteration T (t =T) |
|  | Weights |
| Case 1 | Less weight |
| Case 2 | More weight |
| Case 3 | Less weight |
| … |  |
| Case m | More weight |
| CT:hT | |

|  |  |
| --- | --- |
|  | Iteration 2 (t =2) |
|  | Weights |
| Case 1 | Less weight |
| Case 2 | More weight |
| Case 3 | Less weight |
| … |  |
| Case m | More weight |
| C2:h2 | |

|  |  |  |
| --- | --- | --- |
|  | Iteration 1 (t = 1) | |
|  | Weights | Result |
| Case 1 | 1/m | Correctly classified |
| Case 2 | 1/m | Misclassified |
| Case 3 | 1/m | Correctly classified |
| … | … |  |
| Case m | 1/m | Misclassified |
| C1:h1 | | |

Reweight

Reweight

,

### **Toy example**

* Initial set-up

|  |  |  |  |
| --- | --- | --- | --- |
| Initial set up | | | |
| X1 | X2 | Y (Actual class) | Weights D1 |
| 1 | 5 | Positive | 0.1 |
| 2 | 3 | Positive | 0.1 |
| 3 | 2 | Negative | 0.1 |
| 4 | 6 | Negative | 0.1 |
| 4 | 7 | Positive | 0.1 |
| 5 | 9 | Positive | 0.1 |
| 6 | 5 | Negative | 0.1 |
| 6 | 7 | Positive | 0.1 |
| 8 | 5 | Negative | 0.1 |
| 8 | 8 | Negative | 0.1 |
|  |  |  | 1 |

* Error calculation: There are **three cases misclassified**: Hence error = 0.30
* Classifier weight calculation:
* Calculating the weights for the cases:
* On dividing by the normalization factor, we get
* Above steps are repeated for other cases. At the end we get,

|  |  |  |
| --- | --- | --- |
| Round 1 | | |
| h1e | Error | D2 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 1 | 0.1 | 0.17 |
| 1 | 0.1 | 0.17 |
| 0 | 0 | 0.07 |
| 1 | 0.1 | 0.17 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |

* Same process is repeated in subsequent iterations.

### **Choosing the number of iterations**

* Plot the error rate and number of iterations. Using the scree plot, choose the value after which the error rate does not decrease.
* In the above example, 80 iterations can be chosen.

### **Margin**

* Difference between weighted fraction of classifiers voting correctly and the fraction corresponding to those voting incorrectly
* Range for Margin [-1, 1]
  + When Margin = 1, then all classifiers are correct and hence it is the ideal scenario
  + When Margin = -1, all classifiers are incorrect and hence the model is poor in performance

### **Applying Weights**

* Gini index calculation for Decision trees:
* For the above node, Gini index is calculated as follows:

= 1 – [

* Gini index for the child node is calculated in the same fashion. The difference between Gini index of parent and child nodes is calculated. If the difference is greater than a threshold value, then the parent node will be split.
* In the above case, counts are used to calculate the Gini index. In Adaboost, weights are used instead:

|  |  |  |
| --- | --- | --- |
| h1e | Error | Weights |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |
| 1 | 0.1 | 0.17 |
| 1 | 0.1 | 0.17 |
| 0 | 0 | 0.07 |
| 1 | 0.1 | 0.17 |
| 0 | 0 | 0.07 |
| 0 | 0 | 0.07 |

## **Stacking**

* Uses *meta* *learner* instead of voting to combine predictions of base learners
* Predictions of base learners (*level-0 models*) are used as input for meta learner (*level-1 model)*

**META**

**LEARNER**

Level – 0 models

Meta instance

Level – 1 models

– Prediction for instance x by classifier.

– Final prediction