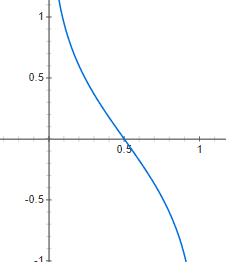
Dans la partie de boosting, nous cherchons une combinaison linéaire des classifieurs faibles entrainés dans la première partie.

Pour cela nous allons itérer à chaque fois sur toutes les images et tous nos classifieurs pour déterminer le classifieur avec la plus faible erreur pondérée. Pour faire cela, nous avons utilisé MPI\_MINLOC pour déterminer l’indice du meilleur classifieur. A chaque étape, nous avons ensuite mis à jour les poids des images pour rendre moins efficace notre classifieur (on a baissé le poids des images que ce classifieur classait déjà correctement) afin de sélectionner d’autres classifieurs par la suite.

L’idée est de finalement obtenir une combinaison de classifieurs qui sont efficaces sur l’ensemble des images.

Pour faire cela, nous avons utilisé la fonction \frac{1}{2}\log(\frac[1-x}{x}) pour récupérer une pondération en fonction de l’erreur pondérée du classifieur.



On voit que cette fonction change de signe autour d’une erreur de 0.5 et cela a conduit à la limitation du nombre de classifieur faible sélectionné par notre algorithme. En effet, un classifieur avec une erreur un peu plus faible était sélectionné puis à cause de la modification des poids, il devenait moins efficace et passait à une erreur supérieure à 0.5 et donc lorsqu’il était repris, on améliorait cette fois ses performances, le faisant à nouveau passer sous la barre de 0.5 d’erreur pondérée. Cette apparition de cycle nous a empêché d’obtenir une combianaison riche de classifieur puisqu’en général on obtenant moins d’une dizaine de classifieur indépendamment de la valeur de N.

Enfin, notre classifieur final était très peu sensible à \theta.

Les paramètres K=2000, epsilon = 0.1, n = 200, N = 20, theta = 0 et une normalisation de 0.001 nous ont permis d’obtenir un F-score de 0.444 sur toutes les images du dossier test.

On a cependant été limité dans l’apprentissage par lae nombre d’images disponibles et notamment le fait qu’elles soient identiques dans les 3 doisseirs (app, test et dev). On aurait peut-être du passer à des caractéristiques rectangulaires pour obtenir plus d’informations. Enfin, on pourrait envisager de diviser la valeur de chaque caractéristique par sa taille afin de normaliser toutes les valeurs afin de pouvoir les comparer entre elles et qu’elles soient toutes de l’ordre de l’unité.