Problem Set-3

Problem #1: VC Dimension

1. A binary classification problem in R^3

Hypothesis space - axis aligned 3-d boxes
'+' inside '-' outside

Consider any 7 distinct points in R^3. Now consider a 3-d cube with minimum and maximum x, y and z coordinate.

Therefore, we can say that the number of points closest to each side will be atmost six. Now let all these points be '+' labelled and the seventh point is labelled '-'. Therefore this set of points i.e. 7 distinct points cannot be shattered.

The same argument can be used to show that any number of distinct points greater than 7 are not shattered by this hypothesis so we can say; the VC dimension is atleast 6.

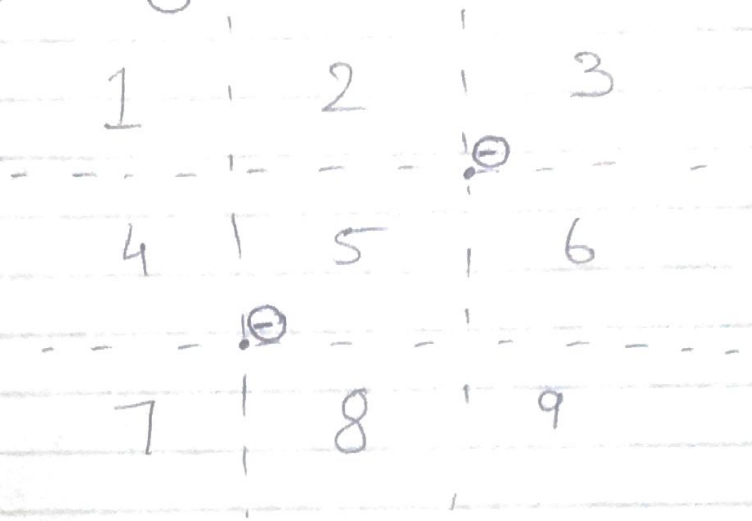
We can generalize the argument to axis aligned boxes in R^d as the VC dimension is equal to the number of sides of the box.
That is 2^d for R^d.

$\therefore$ $R^2 = 4$ $R^3 = 6$ $R^4 = 8$ $R^5 = 10$.

2. A binary classification problem in R^2

Hypothesis space - pair of axis alligned rectangles '+' inside '-' outside

For some points to not get shattered by the hypothesis ie. pair of axis alligned rectangles. the following situation should occur.

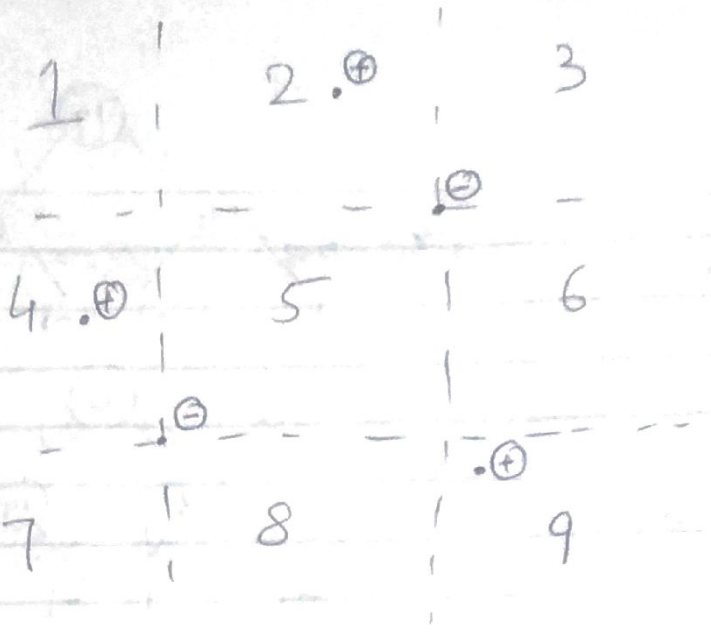


The nine regions are marked as per the mobile keypad format.

If there are positive '+' points in the following area.

1) 1, 5, 9 2) 1, 6, 8 3) 2, 4, 9 4) 2, 6, 7 5) 3, 4, 8
6) 3, 5, 7

In other words take 3) 2, 4, 9 case. Which is drawn in the following diagram



The above data points can't be shattered by the pair of axis parallel rectangle.

So the VC dimension of this hypothesis is the ~~maximum~~ minimum number of data points that when taken distinctally should fall into any of the six pattern of area.

So, the number number of data points that are needed to fall into any of the six pattern of area for not shattering is 11. 9 areas + 2 points to mark the areas.

∴ the VC dimension is 10.

Samples needs to sufficient to guarantee that an optimal learning algorithm will attain an accuracy of 0.8 with probability at least .95

$0.95 = 1 - \delta \Rightarrow \delta = 0.05$

ϵ = error of the hypothesis ∴ $1 - \epsilon = 0.8$

$\Rightarrow \epsilon = 0.2$

$$m \geq \frac{1}{\epsilon}\left(4 \cdot \log \frac{2}{\delta} + VC(H) \cdot \log \frac{13}{\epsilon}\right)$$

putting values of ϵ, δ and $VC(H) = 10$ we get

$$m \geq 1743.5$$

∴ Minimum '1744' samples would be sufficient.

Problem 2#: Medical Diagnosis

1. Train a decision tree using ID3.

Depth of learned decision tree = 12

Accuracy of the learned decision tree on the test dataset is 74.33 %.

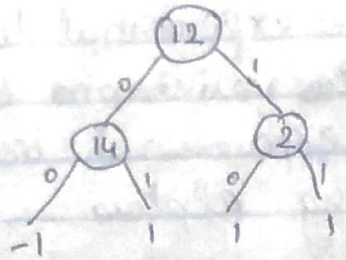
2. Depth 2 decision trees

(a) AdaBoost.

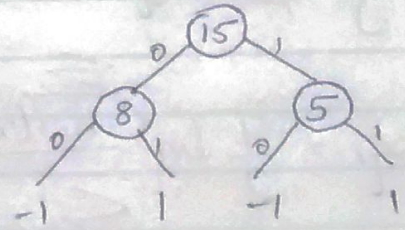
M=5. depth 2 decision tree.

The node number here as I have taken are the column numbers of the training set.
∴ '1' column is class label

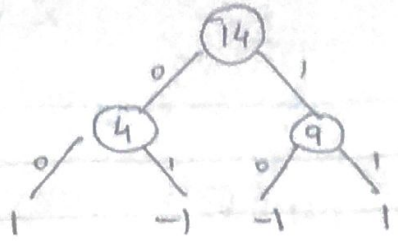
m=1
$\epsilon_1 = 0.2125$ ⟶



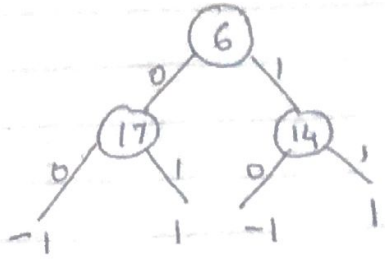
m=2
$\epsilon_2 = 0.3035$ ⟶



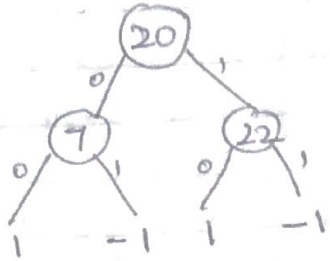
$m = 3$
$\epsilon_3 = 0.3110$



$m = 4$
$\epsilon_4 = 0.3164$



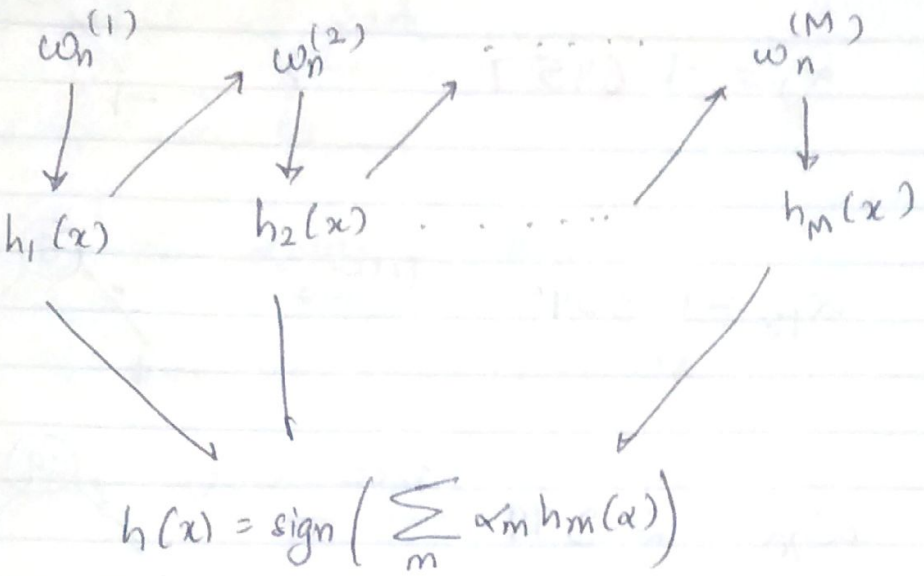
$m = 5$
$\epsilon_5 = 0.3414$



(b)
The accuracy of the AdaBoost classifier on the test data increases for the increase in value of M.

	Accuracy
M = 10	65.2406
M = 20	66.3102
M = 50	66.3102
M = 100	67.3797

AdaBoost is graphically Illustrated as



Algorithmically it is written as

1. Weight all training samples equal
   // as we don't know which will be classify correctly / incorrectly
2. Train model on training set
3. Compute error of model on training set.
   // take the model having minimum error of model
4. Increase weights on training cases model gets wrong
   // this emphases more on misclassified points
5. Train a new model on reweighted training set
6. Repeat.
7. Final model: weighted prediction of each model.

3. Hypothesis space consists only of depth 1 decision tree

(a) Co-Ordinate Descent

Here we take a set of weak learners. In this case it is all depth 1 decision trees.

∴ total numbers of learners in our case is $\underset{\text{no of attributes}}{22} \times 4 = 88$

Then now we minimize the exponential loss which is

$$l(\alpha_1, \ldots, \alpha_T) = \sum_i \exp\left(-y_i \cdot \sum_t \alpha_t h_t(x^{(i)})\right)$$

Each α from 1 to T are assigned for the number of weak learners we have.

As the exponential loss is convex in α_t, we can find the minimum loss, or in other words move the exponential loss to the smaller values by updating alphas.

Solving for $\alpha_{t'}$

$$\alpha_{t'} = \frac{1}{2} \ln \frac{\sum_{i=h_{t'}(x^{(i)})=y_i} \exp\left(-y_i \sum_{t \neq t'} \alpha_t h_t(x^{(i)})\right)}{\sum_{i=h_{t'}(x^{(i)}) \neq y_i} \exp\left(-y_i \cdot \sum_{t \neq t'} \alpha_t h_t(x^{(i)})\right)}$$

Now in co-ordinate descent we initialize some α_ts randomly and decrease the value of exponential loss.

Steps for coordinate descent

1) Initialize the alphas $[1 \ldots 88]$

2) Update α_t keeping $\alpha_{t \neq t'}$ constants.

3) Calculate the exponential loss

4) Repeat if not converged to minimum

The classifier used now to classify the unknown data point is

$$\text{sign}\left(\sum_t \alpha_t h_t(x^{(i)})\right)$$

Implementing the code

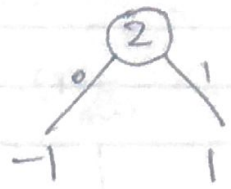
As it takes too long to exactly find the convergance of loss, I stopped iterating when value of loss became less than 40 (which took more than 40 mins).

The alphas at this point were varing both sides of zero.

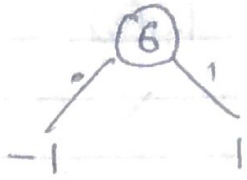
Taking 3 random alphas from [1..88]

attribute indices start from '2'.

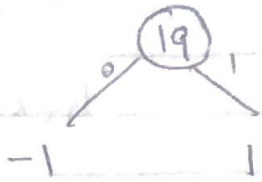
$\alpha_2 = -1.6957$ tree →



$\alpha_{18} = 0.3095$ tree →



$\alpha_{70} = 2.379$ tree →



Value of loss is 39.9998

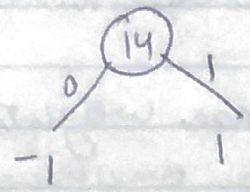
(b) The accuracy on test set is 70.05%

(c) For adaboost with this hypothesis with M = 20

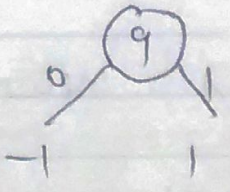
the accuracy on the test set is 52.9412%.

The alphas in this case are

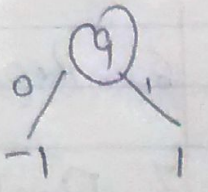
$\alpha_1 = 0.4847$ tree →



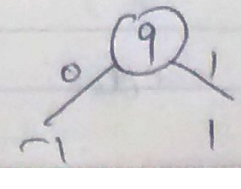
$\alpha_2 = 0.6209$ tree →



$\alpha_3 = 0.8894$ tree →



$\alpha_{20} = 6.6887$ tree →



alphas in adaboost are increasing in each iteration. and they only coreponds to the tree having minimum weight.

Whereas in coordinate descent alphas are for each hypothesis (weak learner) and is updated accordingly to decrease or minimize the exponential loss.