1. parameter estimation

1.1

（1）Assume . Given λ, the possibility of X is

MLE maximize this possibility, such that maximize

Since the

Maximize

we make the derivative equal 0, such that:

Then we got:

It is unbiased, Prove :

(2)

Such that:

(3)

Maximize , maximize

Make the derivative equals 0, then we get

Such that

2. decision tree

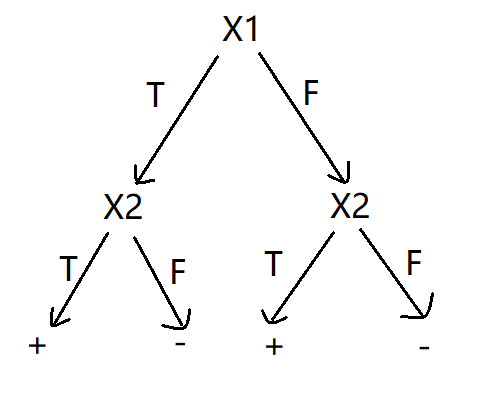
2.1 ID3

(1)

(2) 0.8

0.94

(3)



3

1. When the learning rate is large, the algorithm converge faster than that scenario at which the learning rate is small. But if the learning rate is too large, the algorithm may not converge.
2. Yes. But the training error rate can’t be 0. Because the data set is not linearly separable. The below figure show that even for the feature 1 and feature 2, this data set is not linearly separable.

Now let me explain why this algorithm converge. The training error rate fluctuates around 0.2. Take a randomly split dataset as an example, we record the change of 1/||w||(proportional to the maximum margin), and the change of miss classified instances after each update.

The orange line represents the trend of the 1/||w||, the blue line represents the trend of the number of miss classified instances after each iteration. we can see that, as the update times grow, the 1/||w|| tends to be a constant value, and the miss classified number line is always fluctuating around the horizontal line 0.2. That means the 1/||w|| is not going to decrease. Meanwhile, the number of miss classified instances is still very high and not going to decrease either. Thus the algorithm will converge to a non-zero error rate.

(3) Use logistic function to calculate the possibility that the estimated label equals 1.

In the above figure, the blue point represent the instance whose actual label is 1, the orange point represent the instance whose actual label is 0. And the y value of this point represent the possibility that the label equals 1. We can see from the above figure that, some points that are 100% estimated to be labeled 1, but is actually label 0. That means the model is not quite good. The accuracy is just 21/34 ~= 0.62

(4) Add a new feature: = feature(1) \*feature(1)

The accuracy should increase, because we could use the new feature to get more flexibility and make this model more complex. That means the hyperplane doesn’t need to be a strict line, there could be some variance in this hyperplane.

Run the initial algorithm for three times, we get the accuracy:

0.59

0.74

0.56

Run the improved algorithm for three times, we get the accuracy:

0.65

0.68

0.56

From the mean perspective, the average accuracy of initial algorithm is equal to that of the improved algorithm. I think maybe it is because the number of train instances and test instances are not enough, such that the initial algorithm has higher bias and the improved algorithm has higher variance. That makes them seem equal.