**Machine Learning**

**HW1**

Yun LI 6035800416 yli079@usc.edu

**Problem 1:**

**Solution 1.a**



Because of conditional independence, we can write this:



Because of the following:



We can write this:



Where



And where



**Solution 1.b**

We first calculate parameters µ and σ For general Gaussian Distribution, and then induce it to our problem. The log likelihood function is:



Setting the partial derivative to 0, we have:



Therefore:



In our problem, with Gaussian Naïve Bayes, we must estimate the mean and standard deviation of each of these Gaussians:



Therefore, the maximum likelihood estimator for these parameters is:



Where the superscript j refers to the jth training example, and where is 1 if  and 0 otherwise. It is used to select only those training examples for which .

**Problem 2:**

**Solution 2.a**

We use brute force search

1 Set an increment k (i.e. 0.1, 0.01) and start from 

2 for i = 1 to 1/k :

For j = 1 to 3 :



Compute 

3 compare and find the optimal w for optimal 

**Solution 2.b**

We use some methods similar to coordinate descent. That is:

1 Start with some random values for w=<w1, w2, w3,……, w1000>

2 for i from I to 1000

Find the optimal wi ( fix other weight dimensions). That is,



3 repeat 2 until f no longer changes.

**Problem 3**

**Solution 3.a**



**Solution 3.b**

In the book ( 5th edition, page 278), We computed the Hessian of L(w):



Note that, for any i,  is positive semi-definite, because of :



Since  is always between (0, 1),  (1-) is always positive, which is the coefficient of . So that the Hessian is a sum of positive semi-definite matrices, thus, it is positive semi-definite.

Thus the loss function is convex.

**Solution 3.c**

The decision boundary is determined by. If the training set is linearly separable, any point satisfies when; and any point satisfies when.

If



Then we substitute equations (1) and (2), which makes

.

This means the above equation (1) and (2) determine some w that is optimal.

In sigmoid function, w that satisfies equation (1) and (2) must be those whose magnitude is infinite.

Thus, any step function between the two classes is a maximum likelihood estimate for the loss function. This means that MLE will select w with infinite magnitude, and there exists multiple such w.

**Solution 3.d**

The original derivative for the ith training sample is：



The regularized one is:



Thus, the gradient for the new function with penalty is:



**Solution 3.e**

Because the penalty term is a strictly convex function, the new function with penalty term is strictly convex. To show this, the second-order derivative is

 ()

Thus, the function with regularizer is strictly convex. So the multiple minima case does not happen. We have a unique solution here.

**Problem 4**

**Solution 4.a**

Conditional entropy is

Entropy for weather:



Entropy for traffic:



Because conditional entropy for traffic is less than weather, the information gain for traffic is higher than weather, Thus we choose to first split on traffic (smaller uncertainty).

**Solution 4.b**

T1 and T2 are the same. Since information gain could be written as a sequence of multiply of probabilities. The probabilities will stay the same even after we normalize the data in decision tree. Thus the information gain for each split will be the same. The tree will stay the same.

**Solution 4.c**



is between (0,1), so we need to determine the sign of 



Therefore, it’s a concave function. With one global maximum at the point where



Therefore,



Thus, for any discrete probability distribution , the value of the Gini index is less than or equal to the corresponding value of cross-entropy.

**Problem 5:**

**Solution 5.4**

(1) KNN Performance:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| dataset\k | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 |
| valid | 0.7558 | 0.8046 | 0.8329 | 0.8406 | 0.8689 | 0.88638 | 0.8560 | 0.8278 | 0.8252 | 0.8226 | 0.8098 | 0.8252 |
| test | 0.7943 | 0.8638 | 0.9075 | 0.8920 | 0.8946 | 0.8792 | 0.8792 | 0.8638 | 0.8612 | 0.8535 | 0.8483 | 0.8432 |
| train | 0.7779 | 0.8316 | 0.8663 | 0.8842 | 0.8863 | 0.8905 | 0.8842 | 0.8705 | 0.8589 | 0.8526 | 0.8537 | 0.8458 |

(2)Decision Tree:

Valid set:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Crit\Minleaf | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Gini index | 0.9332 | 0.9332 | 0.9383 | 0.9383 | 0.9409 | 0.9460 | 0.9486 | 0.9512 | 0.9460 | 0.9383 |
| Cross-entropy | 0.9332 | 0.9332 | 0.9383 | 0.9383 | 0.9409 | 0.9460 | 0.9486 | 0.9512 | 0.9460 | 0.9383 |

Test set:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Crit\Minleaf | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Gini index | 0.9434 | 0.9434 | 0.9434 | 0.9434 | 0.9460 | 0.9434 | 0.9460 | 0.9512 | 0.9409 | 0.9409 |
| Cross-entropy | 0.9486 | 0.9486 | 0.9486 | 0.9486 | 0.9512 | 0.9486 | 0.9512 | 0.9512 | 0.9409 | 0.9409 |

Train set

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Crit\Minleaf | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Gini index | 0.9674 | 0.9674 | 0.9653 | 0.9653 | 0.9632 | 0.9621 | 0.9600 | 0.9589 | 0.9526 | 0.9474 |
| Cross-entropy | 0.9705 | 0.9705 | 0.9684 | 0.9684 | 0.9663 | 0.9653 | 0.9632 | 0.9589 | 0.9526 | 0.9474 |



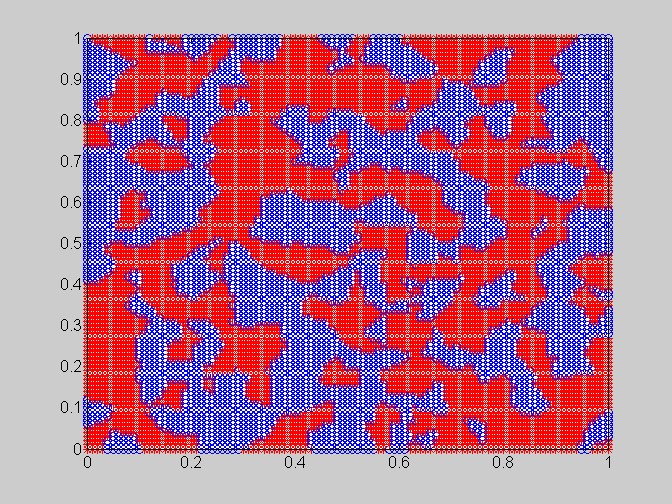
(3)Naive Bayes

|  |  |
| --- | --- |
| Dataset | accuracy |
| Valid data | 0.8380 |
| Test data | 0.8380 |
| Training data | 0.8705 |

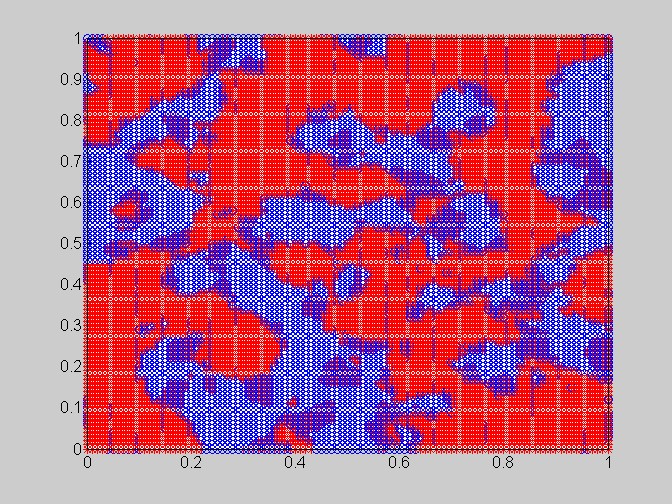
(4)Logistic Regression

|  |  |
| --- | --- |
| Dataset | accuracy |
| Valid data | 0.9152 |
| Test data | 0.9152 |
| Training data | 0.9453 |

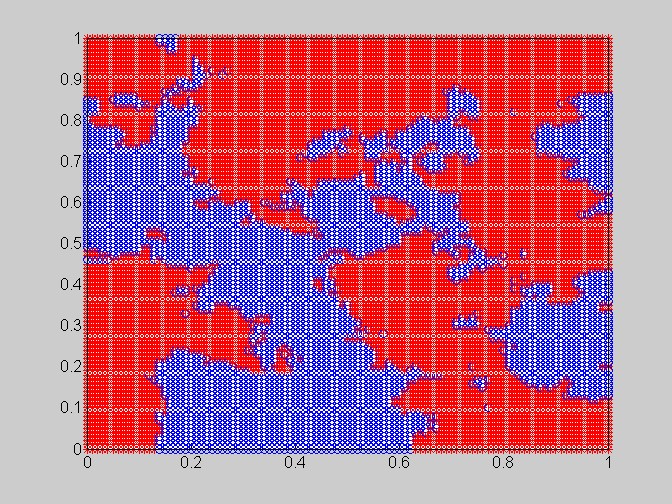
**Solution 5.5**



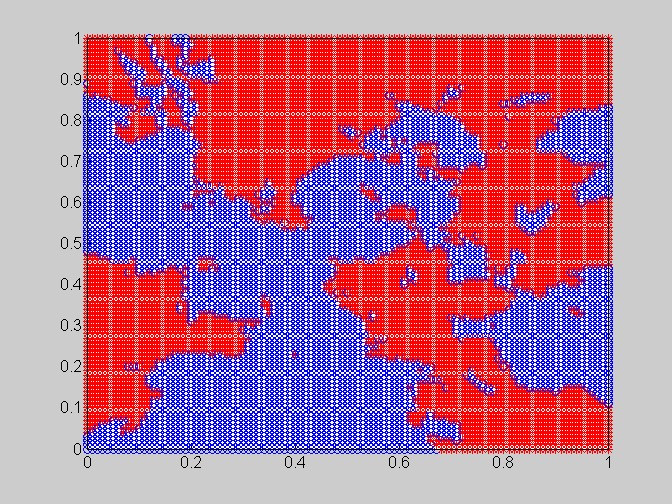
K=1



K=5



K=15



K=20

We can see, as K grows, the decision boundary becomes more smooth. When k is small, we run the risk of being overfitting. The points are really dense, there’s a reasonable chance that there will be noisy data points that are close enough to each other to outvote the correct data points in some region. When k grows, we end up smoothing things out too much and eliminating some important details in the distribution.

Discuss partner : Jieci Liang