Assignment #5

Rajati

Muxin Liang

Contents

Exercise # 1 **3**
(a) Data set . . . 3
(b) Supervised Learning . . . 3
(c) Semisupervied . . . 4
(d) Unsupervised . . . 5
(e) Compare the results . . . 6

Exercise # 2 **7**
(a) Download dataset, choose k . . . 7
(b)Determine family . . . 7
(c)Calculating hamming distance . . . 7

Exercise # 3 **7**

Exercise # 1

Supervised, Semi-supervised and Unsupervised Learning.

(a) Data set

Download the Blood Transfusion Service Center Data Set. Use 20% of positive and negative as test and rest as train.

(b) Supervised Learning

Used L1-SVM to do classification on normalized data. Detailed code in Jupyter Notebook.
As shown in notebook, when using train data as test, we observe 126 "1"s are predicted to 0(False negative) and 7 "0"s are predicted to 1(False Positive). And the overall accuracy is 0.7776.
The figure below shows the ROC and AUC of trainset.

$$CM_{Reference,Prediction} = \begin{pmatrix} & 0 & 1 \\ 0 & 499 & 126 \\ 1 & 7 & 16 \end{pmatrix}$$

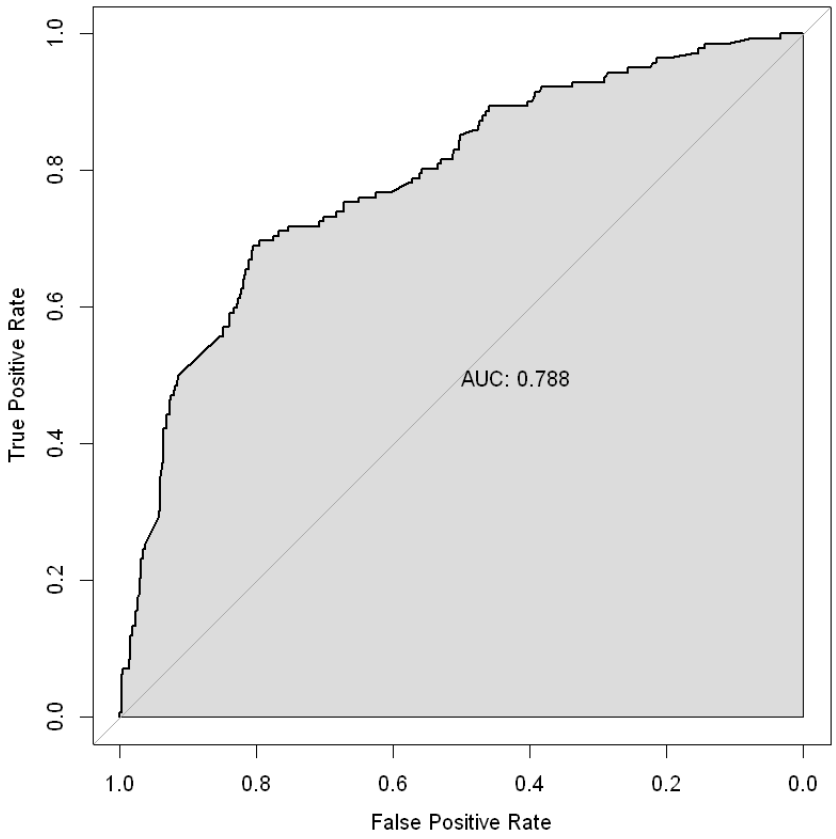


Figure 1: ROC and AUC of trainset

When using test data as test, we observed 34 "1"s are predicted to 0(False negative) and 0 "0"s are predicted to 1(False Positive). And the overall accuracy is 0.7733.
The figure below shows the ROC and AUC of testset.

$$CM_{Reference,Prediction} = \begin{pmatrix} & 0 & 1 \\ 0 & 114 & 34 \\ 1 & 0 & 2 \end{pmatrix}$$

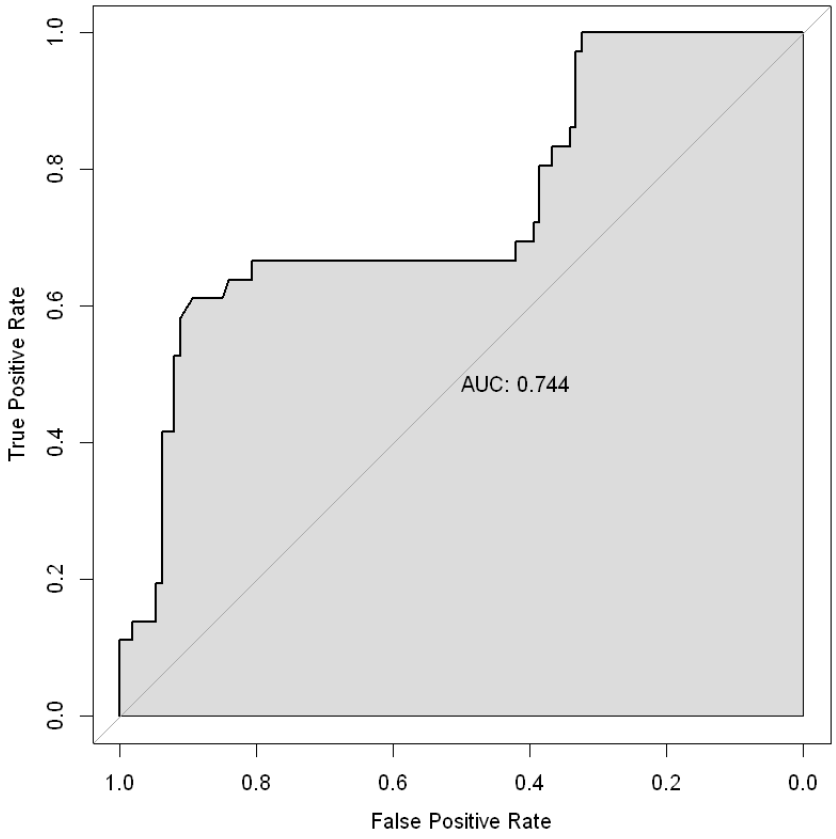


Figure 2: ROC and AUC of trainset

(c) Semisupervied

Select 50% of the positive class along with 50% of the negative class in the training set as labeled data and the rest as unlabelled data. Then we train an L1 SVM on labeded data.
And we find points that closest to decision boundary of SVM and label it by SVM then continue this process until all points used.
Finally we generated RUC, AUC and confusion matrix on test data.
When using test data as test, we observed 31 "1"s are predicted to 0(False negative) and 3 "0"s are predicted to 1(False Positive). And the overall accuracy is 0.7733.

$$CM_{Reference,Prediction} = \begin{pmatrix} & 0 & 1 \\ 0 & 114 & 31 \\ 1 & 3 & 5 \end{pmatrix}$$

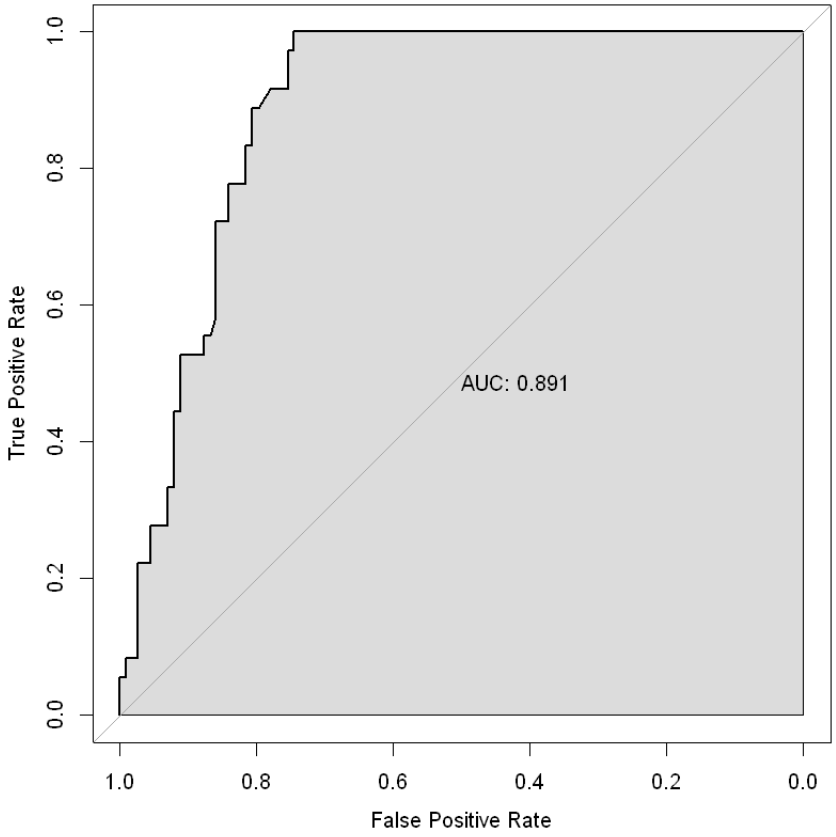


Figure 3: ROC and AUC of trainset

(d) Unsupervised

1. How to make sure algorithm was not trapped? Use the parameter "nstart" in R, which given the number of random assignment in clustering. A large enough nstart will avoid trapping.

2. Detailed clustering:
   Run kmeans on traindata without labels. K=2.

   Then use majority polling for 30 points on clusters and decide the labels.

   Then use the clusters to do classification on testset.

   According to the cluster results, we have to label both cluster as 0. This shows that clustering with k=2 did not given a good result(maybe a larger k will give better result?)

   Finally we generated confusion matrix on train data.

   When using train data as test, we observed 142 "1"s are predicted to 0(False negative) and 0 "0"s are predicted to 1(False Positive). And the overall accuracy is 0.7625.

   $$CM_{Reference,Prediction} = \begin{pmatrix} & 0 & 1 \\ 0 & 456 & 142 \\ 1 & 0 & 0 \end{pmatrix}$$

3. Classification on test using previous result:

When using test data as test, we observed 36 "1"s are predicted to 0(False negative) and 0 "0"s are predicted to 1(False Positive). And the overall accuracy is 0.76.

$$CM_{Reference,Prediction} = \begin{pmatrix} & 0 & 1 \\ 0 & 114 & 36 \\ 1 & 0 & 0 \end{pmatrix}$$

(e) Compare the results

The accuracy of semi-supervised and supervised looks much similar in test set. And unsupervised learning using k=2 means gives bad results because it predict all classes as 0.

Exercise # 2

K-Means Clustering on a Multi-Class and Multi-Label Data Set

(a) Download dataset, choose k

I used CH in choosing the best k, since the data has multiple classes, I choose k start from 5 and use the formula:

$$CH = \frac{B(K)/(k-1)}{W(K)/(n-k)}$$

And choose the maximum CH.
The result shows that the best k is k =8.
Detailed code in Jupyter Notebook.

(b)Determine family

I use majority polling for 30 points that are nearest to the cluster's centers.
Detailed code in Jupyter Notebook.

(c)Calculating hamming distance

I use majority polling for 30 points that are nearest to the cluster's centers.
Then used labeled clusters to predict all of the data.

For every difference in every class, hamming distance plus 1.
Then the average distance is calculated by:

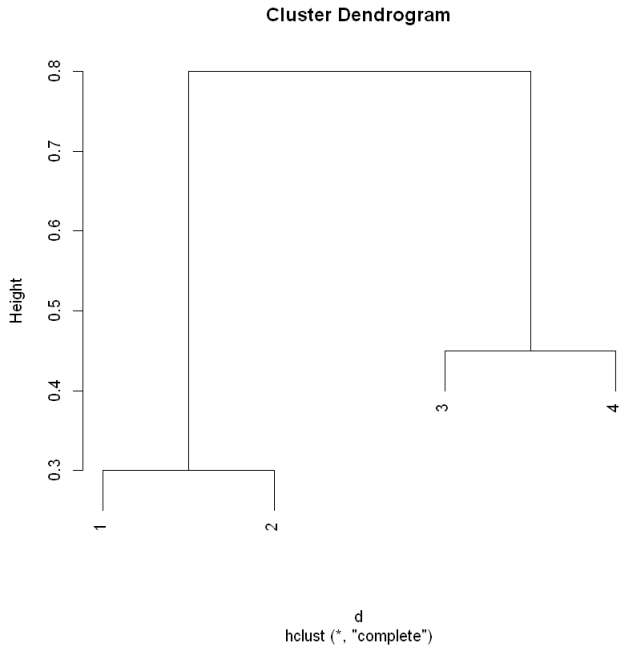
$$avghamming = \frac{HD}{\#ofallclasses}$$

Average hamming distance = 0.1989.
Detailed code in Jupyter Notebook.

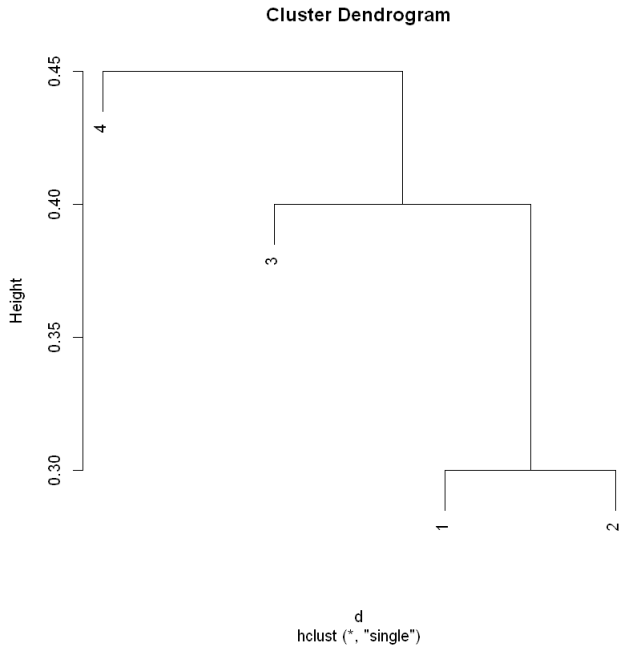
Exercise # 3

ISLR 10.7.2

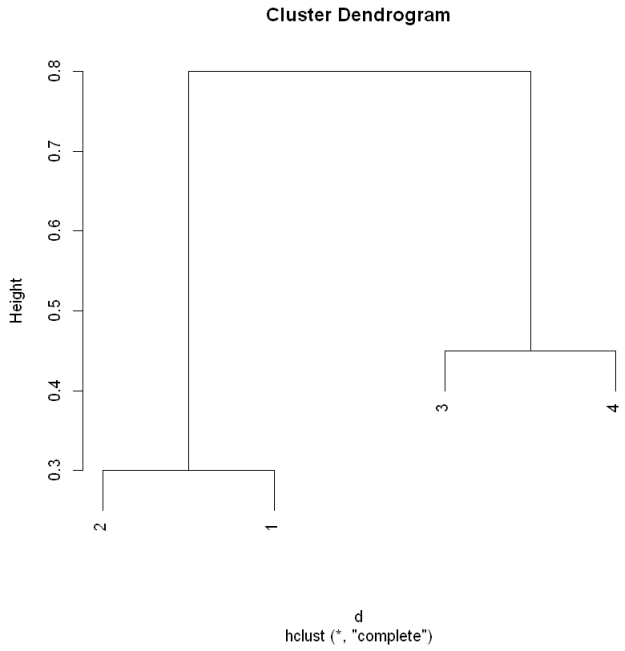
1. Sketch dendrogram using complete linkage



2. Sketch dendrogram using single linkage



3. complete linkage: (1,2) and (3,4)
4. single linkage: (1,2,3) and (4)
5. replaced dendrogram:


Cluster Dendrogram
Height
0.3
0.4
0.5
0.6
0.7
0.8
2
1
3
4
d
hclust (*, "complete")