Carbon

1. Load data

(can only deal with continuous data)

|  |
| --- |
| #load module data  import data  #create a dataset object and read data from file sample.txt  sample = data.DataSet()  sample.read('sample.txt')  #view the data  sample.x  array([[ 5., 1., 1., ..., 3., 1., 1.],  [ 5., 4., 4., ..., 3., 2., 1.],  [ 3., 1., 1., ..., 3., 1., 1.],  ...,  [ 5., 10., 10., ..., 8., 10., 2.],  [ 4., 8., 6., ..., 10., 6., 1.],  [ 4., 8., 8., ..., 10., 4., 1.]])  #view class labels  sample.y[:10]  ['2', '2', '2', '2', '2', '4', '2', '2', '2', '2']  #view dataset dimension  sample.dim()  (699, 9)  #view features  sample.label  ['Clump Thickness', 'Uniformity of Cell Size', 'Uniformity of Cell Shape', 'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin', 'Normal Nucleoli', 'Mitoses']  #view subject ids  sample.key[:5]  ['1000025', '1002945', '1015425', '1016277', '1017023']  #create train dataset and test dataset using 1:10 hold out  train,test = data.holdOut(sample,0.1) |

2. kNN

(this kNN is using Euclidean distance as default distance function)

|  |
| --- |
| #load kNN module  import kNN  #create a classifier  clf = kNN.build()  #train the classifier with train data and k=4  clf.train(train,4)  #to view the model by plot 2 features with class label  clf.view('Uniformity of Cell Size','Uniformity of Cell Shape')  Macintosh HD:Users:wangjiayu:Desktop:Screen Shot 2016-12-30 at 16.59.17.png  #to classify a new subject  clf.classify([5, 4, 4, 5, 7, 10, 3, 2, 1])  '4'  #test the classifier with test data  #and save the predicted labels in result  result = clf.test(test)  the total error rate is: 0.130435  result  ['2', '4', '2', '4', '2', '4', '2', '2', '2', '2', '2', '2', '2', '2', '4', '2', '2', '2', '4', '2', '4', '4', '2', '2', '2', '4', '2', '2', '2', '2', '2', '2', '4', '2', '2', '2', '4', '2', '4', '4', '4', '4', '4', '4', '4', '2', '4', '2', '2', '4', '4', '2', '4', '4', '4', '4', '4', '4', '4', '2', '4', '2', '4', '2', '2', '4', '2', '4', '4']  #to save the classifier in folder models/ as model.knn  clf.save('model')  #to load saved classifier  clf2 = kNN.load('model') |

3. ID3

(works only on nominal data)

|  |
| --- |
| #convert continuous data to nominal data  train.num2nom()  test.num2nom()  train.type  'nominal'  #load ID3 module  import ID3  #create a classifier  clf = ID3.build()  #train the classifier with train data to build the tree  clf.train(train)  #to view the model  clf.view()  Macintosh HD:Users:wangjiayu:Desktop:figure_1.png  #to classify a new subject  clf.classify(['5.0', '4.0', '4.0', '5.0', '7.0', '10.0', '3.0', '2.0', '1.0'])  '4'  #test the classifier with test data  #and save the predicted labels in result  result = clf.test(test)  the total error rate is: 0.115942  result  ['2', '4', '2', '4', '2', '4', '4', '2', '2', '2', '2', '2', '4', '2', '4', '4', '2', '2', '4', '2', '4', '4', '2', '2', '2', '4', '2', '2', '2', '2', '2', '2', '4', '2', '2', '2', '4', '2', '4', '4', '2', '2', '4', '4', '4', '2', '4', '2', '2', '4', '4', '4', '4', '4', '4', '4', '4', '2', '4', '4', '4', '2', '4', '2', '2', '2', '2', '4', '4']  #to save the classifier in folder models/ as model.id3  clf.save('model')  #to load saved classifier  clf3 = ID3.load('model') |

3. NB

|  |
| --- |
| #load NB module  import NB  #create a classifier  clf = NB.build()  #train the classifier with train data to build the tree  clf.train(train)  #to view the model  clf.view()  #to classify a new subject  clf.classify([5, 4, 4, 5, 7, 10, 3, 2, 1])  '4'  #test the classifier with test data  #and save the predicted labels in result  result = clf.test(test)  the total error rate is: 0.043478  result  ['2', '4', '2', '4', '2', '4', '2', '2', '2', '2', '2', '2', '4', '2', '4', '4', '2', '2', '4', '2', '4', '4', '2', '4', '2', '4', '2', '2', '2', '2', '2', '2', '4', '2', '2', '2', '4', '2', '4', '4', '4', '4', '4', '4', '4', '2', '4', '2', '2', '4', '4', '4', '4', '4', '4', '4', '4', '4', '4', '4', '4', '2', '4', '4', '2', '4', '2', '4', '4']  #to save the classifier in folder models/ as model.id3  clf.save('model')  #to load saved classifier  clf4 = NB.load('model') |

|  |
| --- |
| #fast start  import data  sample = data.DataSet()  sample.read('sample.txt')  train,test = data.holdOut(sample,0.1)  reload(kNN)  clf = kNN.build()  clf.train(train,4)  clf.view('Uniformity of Cell Size','Uniformity of Cell Shape')  clf.test(test)  reload(ID3)  clf = ID3.build()  clf.train(train)  clf.classify(['5.0', '4.0', '4.0', '5.0', '7.0', '10.0', '3.0', '2.0', '1.0'])  clf.test(test)  treePlotter.createPlot(train.tree) |