

```

1  """
2  Synthetic Dataset 1
3  Name: Haolun Cheng
4  USCID: 1882563827
5  EE559 HW1
6  """
7
8  import csv
9  import numpy as np
10 import matplotlib.pyplot as plt
11 from plotDecBoundaries import plotDecBoundaries
12
13 clas1xtotal = []
14 clas1yttotal = []
15 clas2xtotal = []
16 clas2yttotal = []
17 allclassdatapoints = []
18 allclasslabels = []
19 allclassmeans = []
20
21 # Open train csv file for training the classifier
22 with open('synthetic1_train.csv', 'r') as train:
23     training_set = csv.reader(train)
24
25     # Train
26     for line in training_set:
27         x, y, label = line[0], line[1], line[2]
28         allclassdatapoints.append((float(x), float(y)))
29         allclasslabels.append(label)
30         if label == '1':
31             clas1xtotal.append(float(x))
32             clas1yttotal.append(float(y))
33         elif label == '2':
34             clas2xtotal.append(float(x))
35             clas2yttotal.append(float(y))
36
37 train.close()
38
39 # Compute mean for each class
40 clas1xmean = np.mean(clas1xtotal)
41 clas1ymean = np.mean(clas1yttotal)
42
43 clas2xmean = np.mean(clas2xtotal)
44 clas2ymean = np.mean(clas2yttotal)
45
46 clas1_mean_point = np.array((clas1xmean, clas1ymean))
47 clas2_mean_point = np.array((clas2xmean, clas2ymean))
48
49 # Variables for plotDecBoundaries
50 allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean]]
51 datapoints = np.array(allclassdatapoints).astype(float)
52 claslabels = np.array(allclasslabels).astype(float)
53 samplemeans = np.array(allclassmeans).astype(float)
54
55 # Classify data points (training set)
56 countTrainingError = 0
57 totalTrainingPoints = 0
58 with open('synthetic1_train.csv', 'r') as training:
59     train_set = csv.reader(training)

```

```

60
61     for line in train_set:
62         totalTrainingPoints += 1
63         x, y, label = line[0], line[1], line[2]
64         trainPoint = np.array((x, y))
65         dist1 = np.linalg.norm(trainPoint.astype(float) - clas1_mean_point)
66         dist2 = np.linalg.norm(trainPoint.astype(float) - clas2_mean_point)
67
68         if dist1 < dist2:
69             if label != '1':
70                 countTrainingError += 1
71         elif dist1 > dist2:
72             if label != '2':
73                 countTrainingError += 1
74
75 training.close()
76
77 # Fine error rate for training set
78 error_rate = float(countTrainingError) / float(totalTrainingPoints)
79 print("Error rate for the training set: " + str(error_rate))
80
81 # Classify data points (test set)
82 countTestError = 0
83 totalTestPoints = 0
84 with open('synthetic1_test.csv', 'r') as test:
85     test_set = csv.reader(test)
86
87     for line in test_set:
88         totalTestPoints += 1
89         x, y, label = line[0], line[1], line[2]
90         testPoint = np.array((x, y))
91         dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
92         dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
93
94         if dist1 < dist2:
95             if label != '1':
96                 countTestError += 1
97         elif dist1 > dist2:
98             if label != '2':
99                 countTestError += 1
100
101 test.close()
102
103 # Fine error rate for test set
104 error_rate = float(countTestError) / float(totalTestPoints)
105 print("Error rate for the test set: " + str(error_rate))
106
107 # Plot the data points
108 xAxis = [i[0] for i in allclassdatapoints]
109 yAxis = [j[1] for j in allclassdatapoints]
110 plt.plot(xAxis, yAxis, 'r.')
111 plt.xlabel('Feature1')
112 plt.ylabel('Feature2')
113 plt.title('Feature Plot of all Elements')
114 plt.show()
115
116 # Plot the decision boundaries
117 plotDecBoundaries(datapoints, claslabels, samplemeans)

```

```

1  """
2  Synthetic Dataset 2
3  Name: Haolun Cheng
4  USCID: 1882563827
5  EE559 HW1
6  """
7
8  import csv
9  import numpy as np
10 import matplotlib.pyplot as plt
11 from plotDecBoundaries import plotDecBoundaries
12
13 clas1xtotal = []
14 clas1yttotal = []
15 clas2xtotal = []
16 clas2yttotal = []
17 allclassdatapoints = []
18 allclasslabels = []
19 allclassmeans = []
20
21 # Open train csv file for training the classifier
22 with open('synthetic2_train.csv', 'r') as train:
23     training_set = csv.reader(train)
24
25     # Train
26     for line in training_set:
27         x, y, label = line[0], line[1], line[2]
28         allclassdatapoints.append((float(x), float(y)))
29         allclasslabels.append(label)
30         if label == '1':
31             clas1xtotal.append(float(x))
32             clas1yttotal.append(float(y))
33         elif label == '2':
34             clas2xtotal.append(float(x))
35             clas2yttotal.append(float(y))
36
37 train.close()
38
39 # Compute mean for each class
40 clas1xmean = np.mean(clas1xtotal)
41 clas1ymean = np.mean(clas1yttotal)
42
43 clas2xmean = np.mean(clas2xtotal)
44 clas2ymean = np.mean(clas2yttotal)
45
46 clas1_mean_point = np.array((clas1xmean, clas1ymean))
47 clas2_mean_point = np.array((clas2xmean, clas2ymean))
48
49 # Variables for plotDecBoundaries
50 allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean]]
51 datapoints = np.array(allclassdatapoints).astype(float)
52 claslabels = np.array(allclasslabels).astype(float)
53 samplemeans = np.array(allclassmeans).astype(float)
54
55 # Classify data points (training set)
56 countTrainingError = 0
57 totalTrainingPoints = 0
58 with open('synthetic2_train.csv', 'r') as training:
59     train_set = csv.reader(training)

```

```

60
61     for line in train_set:
62         totalTrainingPoints += 1
63         x, y, label = line[0], line[1], line[2]
64         trainPoint = np.array((x, y))
65         dist1 = np.linalg.norm(trainPoint.astype(float) - clas1_mean_point)
66         dist2 = np.linalg.norm(trainPoint.astype(float) - clas2_mean_point)
67
68         if dist1 < dist2:
69             if label != '1':
70                 countTrainingError += 1
71         elif dist1 > dist2:
72             if label != '2':
73                 countTrainingError += 1
74
75 training.close()
76
77 # Fine error rate for training set
78 error_rate = float(countTrainingError) / float(totalTrainingPoints)
79 print("Error rate for the training set: " + str(error_rate))
80
81 # Classify data points (test set)
82 countTestError = 0
83 totalTestPoints = 0
84 with open('synthetic2_test.csv', 'r') as test:
85     test_set = csv.reader(test)
86
87     for line in test_set:
88         totalTestPoints += 1
89         x, y, label = line[0], line[1], line[2]
90         testPoint = np.array((x, y))
91         dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
92         dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
93
94         if dist1 < dist2:
95             if label != '1':
96                 countTestError += 1
97         elif dist1 > dist2:
98             if label != '2':
99                 countTestError += 1
100
101 test.close()
102
103 # Fine error rate for test set
104 error_rate = float(countTestError) / float(totalTestPoints)
105 print("Error rate for the test set: " + str(error_rate))
106
107 # Plot the data points
108 xAxis = [i[0] for i in allclassdatapoints]
109 yAxis = [j[1] for j in allclassdatapoints]
110 plt.plot(xAxis, yAxis, 'r.')
111 plt.xlabel('Feature1')
112 plt.ylabel('Feature2')
113 plt.title('Feature Plot of all Elements')
114 plt.show()
115
116 # Plot the decision boundaries
117 plotDecBoundaries(datapoints, claslabels, samplemeans)

```

```

1  """
2  Wine Dataset (for question (c))
3  Name: Haolun Cheng
4  USCID: 1882563827
5  EE559 HW1
6  """
7
8  import csv
9  import numpy as np
10 import matplotlib.pyplot as plt
11 from plotDecBoundaries import plotDecBoundaries
12
13 clas1xtotal = []
14 clas1yttotal = []
15 clas2xttotal = []
16 clas2yttotal = []
17 clas3xttotal = []
18 clas3yttotal = []
19 allclasstwowefeatures = []
20 allclasslabels = []
21 allclassmeans = []
22 feature1 = []
23 feature2 = []
24 feature3 = []
25
26 # Open train csv file for training the classifier
27 with open('wine_train.csv', 'r') as train:
28     training_set = csv.reader(train)
29
30     # Train
31     for line in training_set:
32         x, y, label = line[0], line[1], line[-1]
33         allclasstwowefeatures.append((float(x), float(y)))
34         allclasslabels.append(label)
35         if label == '1':
36             clas1xttotal.append(float(x))
37             clas1yttotal.append(float(y))
38             feature1.append((float(x), float(y)))
39         elif label == '2':
40             clas2xttotal.append(float(x))
41             clas2yttotal.append(float(y))
42             feature2.append((float(x), float(y)))
43         else:
44             clas3xttotal.append(float(x))
45             clas3yttotal.append(float(y))
46             feature3.append((float(x), float(y)))
47
48 train.close()
49
50 # Compute mean for each class
51 clas1xmean = np.mean(clas1xttotal)
52 clas1ymean = np.mean(clas1yttotal)
53
54 clas2xmean = np.mean(clas2xttotal)
55 clas2ymean = np.mean(clas2yttotal)
56
57 clas3xmean = np.mean(clas3xttotal)
58 clas3ymean = np.mean(clas3yttotal)
59

```

```

60 clas1_mean_point = np.array((clas1xmean, clas1ymean))
61 clas2_mean_point = np.array((clas2xmean, clas2ymean))
62 clas3_mean_point = np.array((clas3xmean, clas3ymean))
63
64 # Variables for plotDecBoundaries
65 allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean],
66                  [clas3xmean, clas3ymean]]
67 twofeatures = np.array(allclasstwofeatures).astype(float)
68 claslabels = np.array(allclasslabels).astype(float)
69 samplemeans = np.array(allclassmeans).astype(float)
70
71 # Classify data points (training set)
72 countTrainingError = 0
73 totalTrainingPoints = 0
74 with open('wine_train.csv', 'r') as training:
75     train_set = csv.reader(training)
76     for line in train_set:
77         totalTrainingPoints += 1
78         x, y, label = line[0], line[1], line[-1]
79         trainPoint = np.array((x, y))
80         dist1 = np.linalg.norm(trainPoint.astype(float) - clas1_mean_point)
81         dist2 = np.linalg.norm(trainPoint.astype(float) - clas2_mean_point)
82         dist3 = np.linalg.norm(trainPoint.astype(float) - clas3_mean_point)
83
84         if dist1 < dist2 and dist1 < dist3:
85             if label != '1':
86                 countTrainingError += 1
87         if dist2 < dist1 and dist2 < dist3:
88             if label != '2':
89                 countTrainingError += 1
90         if dist3 < dist1 and dist3 < dist2:
91             if label != '3':
92                 countTrainingError += 1
93
94 training.close()
95
96 # Fine error rate for training set
97 error_rate = float(countTrainingError) / float(totalTrainingPoints)
98 print("Error rate for the training set: " + str(error_rate))
99
100 # Classify data points (test set)
101 countTestError = 0
102 totalTestPoints = 0
103 with open('wine_test.csv', 'r') as test:
104     test_set = csv.reader(test)
105     for line in test_set:
106         totalTestPoints += 1
107         x, y, label = line[0], line[1], line[-1]
108         testPoint = np.array((x, y))
109         dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
110         dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
111         dist3 = np.linalg.norm(testPoint.astype(float) - clas3_mean_point)
112
113         if dist1 < dist2 and dist1 < dist3:
114             if label != '1':
115                 countTestError += 1
116         if dist2 < dist1 and dist2 < dist3:
117             if label != '2':
118

```

```
119         countTestError += 1
120     if dist3 < dist1 and dist3 < dist2:
121         if label != '3':
122             countTestError += 1
123
124 test.close()
125
126 # Fine error rate for test set
127 error_rate = float(countTestError) / float(totalTestPoints)
128 print("Error rate for the test set: " + str(error_rate))
129
130 # Plot the data points
131 xAxis1 = [i[0] for i in feature1]
132 yAxis1 = [i[1] for i in feature1]
133 xAxis2 = [i[0] for i in feature2]
134 yAxis2 = [i[1] for i in feature2]
135 xAxis3 = [i[0] for i in feature3]
136 yAxis3 = [i[1] for i in feature3]
137 plt.plot(xAxis1, yAxis1, 'r.', xAxis2, yAxis2, 'b^', xAxis3, yAxis3, 'g.')
138 plt.xlabel('Feature1')
139 plt.ylabel('Feature2')
140 plt.title('Feature Plot of all Elements')
141 plt.show()
142
143 # Plot the decision boundaries
144 plotDecBoundaries(twofeatures, claslabels, samplemeans)
```

```

1  """
2  Wine Dataset (for question (d & e))
3  Name: Haolun Cheng
4  USCID: 1882563827
5  EE559 HW1
6  """
7
8  import csv
9  import sys
10 import numpy as np
11 import matplotlib.pyplot as plt
12 from plotDecBoundaries import plotDecBoundaries
13
14 # Two best features selection
15 min_train_error = sys.maxsize
16 best_feature_1 = 0
17 best_feature_2 = 0
18
19 for i in range(13):
20     for j in range(13):
21
22         clas1xtotal = []
23         clas1yttotal = []
24         clas2xtotal = []
25         clas2yttotal = []
26         clas3xtotal = []
27         clas3yttotal = []
28         allclasstwowofeatures = []
29         allclasslabels = []
30         allclassmeans = []
31
32         # Open train csv file for training the classifier
33         with open('wine_train.csv', 'r') as train:
34             training_set = csv.reader(train)
35
36             # Train
37             for line in training_set:
38                 x, y, label = line[i], line[j], line[-1]
39                 allclasstwowofeatures.append((float(x), float(y)))
40                 allclasslabels.append(label)
41                 if label == '1':
42                     clas1xtotal.append(float(x))
43                     clas1yttotal.append(float(y))
44                 elif label == '2':
45                     clas2xtotal.append(float(x))
46                     clas2yttotal.append(float(y))
47                 else:
48                     clas3xtotal.append(float(x))
49                     clas3yttotal.append(float(y))
50
51         train.close()
52
53         # Compute mean for each class
54         clas1xmean = np.mean(clas1xtotal)
55         clas1ymean = np.mean(clas1yttotal)
56
57         clas2xmean = np.mean(clas2xtotal)
58         clas2ymean = np.mean(clas2yttotal)
59

```



```

60     clas3xmean = np.mean(clas3xtotal)
61     clas3ymean = np.mean(clas3yttotal)
62
63     clas1_mean_point = np.array((clas1xmean, clas1ymean))
64     clas2_mean_point = np.array((clas2xmean, clas2ymean))
65     clas3_mean_point = np.array((clas3xmean, clas3ymean))
66
67     # Classify data points (training set)
68     countTrainingError = 0
69     totalTrainingPoints = 0
70     with open('wine_train.csv', 'r') as training:
71         train_set = csv.reader(training)
72
73         for line in train_set:
74             totalTrainingPoints += 1
75             x, y, label = line[i], line[j], line[-1]
76             trainPoint = np.array((x, y))
77             dist1 = np.linalg.norm(trainPoint.astype(float) -
clas1_mean_point)
78             dist2 = np.linalg.norm(trainPoint.astype(float) -
clas2_mean_point)
79             dist3 = np.linalg.norm(trainPoint.astype(float) -
clas3_mean_point)
80
81             if dist1 < dist2 and dist1 < dist3:
82                 if label != '1':
83                     countTrainingError += 1
84             if dist2 < dist1 and dist2 < dist3:
85                 if label != '2':
86                     countTrainingError += 1
87             if dist3 < dist1 and dist3 < dist2:
88                 if label != '3':
89                     countTrainingError += 1
90
91     training.close()
92
93     # Find the two features with minimum errors
94     if countTrainingError < min_train_error:
95         min_train_error = countTrainingError
96         best_feature_1 = i
97         best_feature_2 = j
98
99     # Fine error rate for training set
100 error_rate = float(min_train_error) / float(totalTrainingPoints)
101 print("Error rate for the training set: " + str(error_rate))
102 print("Best feature 1: " + str(best_feature_1 + 1))
103 print("Best feature 2: " + str(best_feature_2 + 1))
104
105 # Classify three classes and find mean for the best two features
106 class1 = []
107 class2 = []
108 class3 = []
109 clas1xtotal = []
110 clas1yttotal = []
111 clas2xtotal = []
112 clas2yttotal = []
113 clas3xtotal = []
114 clas3yttotal = []
115 allclasstwofeatures = []
116 allclasslabels = []

```

```

117 allclassmeans = []
118 with open('wine_train.csv') as classify:
119     threeclasses = csv.reader(classify)
120
121     for k in threeclasses:
122         x, y, label = k[best_feature_1], k[best_feature_2], k[-1]
123         allclasstwofeatures.append((float(x), float(y)))
124         allclasslabels.append(label)
125         if label == '1':
126             class1.append((x, y))
127             clas1xtotal.append(float(x))
128             clas1ytotal.append(float(y))
129         elif label == '2':
130             class2.append((x, y))
131             clas2xtotal.append(float(x))
132             clas2ytotal.append(float(y))
133         else:
134             class3.append((x, y))
135             clas3xtotal.append(float(x))
136             clas3ytotal.append(float(y))
137
138 classify.close()
139
140 # Compute mean for each class
141 clas1xmean = np.mean(clas1xtotal)
142 clas1ymean = np.mean(clas1ytotal)
143
144 clas2xmean = np.mean(clas2xtotal)
145 clas2ymean = np.mean(clas2ytotal)
146
147 clas3xmean = np.mean(clas3xtotal)
148 clas3ymean = np.mean(clas3ytotal)
149
150 clas1_mean_point = np.array((clas1xmean, clas1ymean))
151 clas2_mean_point = np.array((clas2xmean, clas2ymean))
152 clas3_mean_point = np.array((clas3xmean, clas3ymean))
153
154 # Variables for plotDecBoundaries
155 allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean],
156                  [clas3xmean, clas3ymean]]
157 twofeatures = np.array(allclasstwofeatures).astype(float)
158 claslabels = np.array(allclasslabels).astype(float)
159 samplemeans = np.array(allclassmeans).astype(float)
160
161 # Plot the data points
162 xAxis1 = [i[0] for i in class1]
163 yAxis1 = [i[1] for i in class1]
164 xAxis2 = [i[0] for i in class2]
165 yAxis2 = [i[1] for i in class2]
166 xAxis3 = [i[0] for i in class3]
167 yAxis3 = [i[1] for i in class3]
168 plt.plot(xAxis1, yAxis1, 'r.', xAxis2, yAxis2, 'b^', xAxis3, yAxis3, 'g.')
169 plt.xlabel('Feature1')
170 plt.ylabel('Feature2')
171 plt.title('Feature Plot of all Elements')
172 plt.show()
173
174 # Plot the decision boundaries
175 plotDecBoundaries(twofeatures, claslabels, samplemeans)

```

```

176 # Classify data points (test set)
177 countTestError = 0
178 totalTestPoints = 0
179 with open('wine_test.csv', 'r') as test:
180     test_set = csv.reader(test)
181
182     for line in test_set:
183         totalTestPoints += 1
184         x, y, label = line[best_feature_1], line[best_feature_2], line[-1]
185         testPoint = np.array((x, y))
186         dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
187         dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
188         dist3 = np.linalg.norm(testPoint.astype(float) - clas3_mean_point)
189
190         if dist1 < dist2 and dist1 < dist3:
191             if label != '1':
192                 countTestError += 1
193         if dist2 < dist1 and dist2 < dist3:
194             if label != '2':
195                 countTestError += 1
196         if dist3 < dist1 and dist3 < dist2:
197             if label != '3':
198                 countTestError += 1
199
200 test.close()
201
202 # Fine error rate for test set
203 error_rate = float(countTestError) / float(totalTestPoints)
204 print("Error rate for the test set: " + str(error_rate))

```