```python
"""
Synthetic Dataset 1
Name: Haolun Cheng
USCID: 1882563827
EE559 HW1
"""

import csv
import numpy as np
import matplotlib.pyplot as plt
from plotDecBoundaries import plotDecBoundaries

clas1xtotal = []
clas1ytotal = []
clas2xtotal = []
clas2ytotal = []
allclassdatapoints = []
allclasslabels = []
allclassmeans = []

# Open train csv file for training the classifier
with open('synthetic1_train.csv', 'r') as train:
    training_set = csv.reader(train)

    # Train
    for line in training_set:
        x, y, label = line[0], line[1], line[2]
        allclassdatapoints.append((float(x), float(y)))
        allclasslabels.append(label)
        if label == '1':
            clas1xtotal.append(float(x))
            clas1ytotal.append(float(y))
        elif label == '2':
            clas2xtotal.append(float(x))
            clas2ytotal.append(float(y))

train.close()

# Compute mean for each class
clas1xmean = np.mean(clas1xtotal)
clas1ymean = np.mean(clas1ytotal)

clas2xmean = np.mean(clas2xtotal)
clas2ymean = np.mean(clas2ytotal)

clas1_mean_point = np.array((clas1xmean, clas1ymean))
clas2_mean_point = np.array((clas2xmean, clas2ymean))

# Variables for plotDecBoundaries
allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean]]
datapoints = np.array(allclassdatapoints).astype(float)
claslabels = np.array(allclasslabels).astype(float)
samplemeans = np.array(allclassmeans).astype(float)

# Classify data points (training set)
countTrainingError = 0
totalTrainingPoints = 0
with open('synthetic1_train.csv', 'r') as training:
    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)
```

```
"""
Synthetic Dataset 2
Name: Haolun Cheng
USCID: 1882563827
EE559 HW1
"""

import csv
import numpy as np
import matplotlib.pyplot as plt
from plotDecBoundaries import plotDecBoundaries

clas1xtotal = []
clas1ytotal = []
clas2xtotal = []
clas2ytotal = []
allclassdatapoints = []
allclasslabels = []
allclassmeans = []

# Open train csv file for training the classifier
with open('synthetic2_train.csv', 'r') as train:
    training_set = csv.reader(train)

    # Train
    for line in training_set:
        x, y, label = line[0], line[1], line[2]
        allclassdatapoints.append((float(x), float(y)))
        allclasslabels.append(label)
        if label == '1':
            clas1xtotal.append(float(x))
            clas1ytotal.append(float(y))
        elif label == '2':
            clas2xtotal.append(float(x))
            clas2ytotal.append(float(y))

train.close()

# Compute mean for each class
clas1xmean = np.mean(clas1xtotal)
clas1ymean = np.mean(clas1ytotal)

clas2xmean = np.mean(clas2xtotal)
clas2ymean = np.mean(clas2ytotal)

clas1_mean_point = np.array((clas1xmean, clas1ymean))
clas2_mean_point = np.array((clas2xmean, clas2ymean))

# Variables for plotDecBoundaries
allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean]]
datapoints = np.array(allclassdatapoints).astype(float)
claslabels = np.array(allclasslabels).astype(float)
samplemeans = np.array(allclassmeans).astype(float)

# Classify data points (training set)
countTrainingError = 0
totalTrainingPoints = 0
with open('synthetic2_train.csv', 'r') as training:
    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)
```

```
"""
Wine Dataset (for question (c))
Name: Haolun Cheng
USCID: 1882563827
EE559 HW1
"""

import csv
import numpy as np
import matplotlib.pyplot as plt
from plotDecBoundaries import plotDecBoundaries

clas1xtotal = []
clas1ytotal = []
clas2xtotal = []
clas2ytotal = []
clas3xtotal = []
clas3ytotal = []
allclasstwofeatures = []
allclasslabels = []
allclassmeans = []
feature1 = []
feature2 = []
feature3 = []

# Open train csv file for training the classifier
with open('wine_train.csv', 'r') as train:
    training_set = csv.reader(train)

    # Train
    for line in training_set:
        x, y, label = line[0], line[1], line[-1]
        allclasstwofeatures.append((float(x), float(y)))
        allclasslabels.append(label)
        if label == '1':
            clas1xtotal.append(float(x))
            clas1ytotal.append(float(y))
            feature1.append((float(x), float(y)))
        elif label == '2':
            clas2xtotal.append(float(x))
            clas2ytotal.append(float(y))
            feature2.append((float(x), float(y)))
        else:
            clas3xtotal.append(float(x))
            clas3ytotal.append(float(y))
            feature3.append((float(x), float(y)))

train.close()

# Compute mean for each class
clas1xmean = np.mean(clas1xtotal)
clas1ymean = np.mean(clas1ytotal)

clas2xmean = np.mean(clas2xtotal)
clas2ymean = np.mean(clas2ytotal)

clas3xmean = np.mean(clas3xtotal)
clas3ymean = np.mean(clas3ytotal)

```

```
clas1_mean_point = np.array((clas1xmean, clas1ymean))
clas2_mean_point = np.array((clas2xmean, clas2ymean))
clas3_mean_point = np.array((clas3xmean, clas3ymean))

# Variables for plotDecBoundaries
allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean],
[clas3xmean, clas3ymean]]
twofeatures = np.array(allclasstwofeatures).astype(float)
claslabels = np.array(allclasslabels).astype(float)
samplemeans = np.array(allclassmeans).astype(float)

# Classify data points (training set)
countTrainingError = 0
totalTrainingPoints = 0
with open('wine_train.csv', 'r') as training:
    train_set = csv.reader(training)

    for line in train_set:
        totalTrainingPoints += 1
        x, y, label = line[0], line[1], line[-1]
        trainPoint = np.array((x, y))
        dist1 = np.linalg.norm(trainPoint.astype(float) - clas1_mean_point)
        dist2 = np.linalg.norm(trainPoint.astype(float) - clas2_mean_point)
        dist3 = np.linalg.norm(trainPoint.astype(float) - clas3_mean_point)

        if dist1 < dist2 and dist1 < dist3:
            if label != '1':
                countTrainingError += 1
        if dist2 < dist1 and dist2 < dist3:
            if label != '2':
                countTrainingError += 1
        if dist3 < dist1 and dist3 < dist2:
            if label != '3':
                countTrainingError += 1

training.close()

# Fine error rate for training set
error_rate = float(countTrainingError) / float(totalTrainingPoints)
print("Error rate for the training set: " + str(error_rate))

# Classify data points (test set)
countTestError = 0
totalTestPoints = 0
with open('wine_test.csv', 'r') as test:
    test_set = csv.reader(test)

    for line in test_set:
        totalTestPoints += 1
        x, y, label = line[0], line[1], line[-1]
        testPoint = np.array((x, y))
        dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
        dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
        dist3 = np.linalg.norm(testPoint.astype(float) - clas3_mean_point)

        if dist1 < dist2 and dist1 < dist3:
            if label != '1':
                countTestError += 1
        if dist2 < dist1 and dist2 < dist3:
            if label != '2':
```

```
                countTestError += 1
        if dist3 < dist1 and dist3 < dist2:
            if label != '3':
                countTestError += 1

test.close()

# Fine error rate for test set
error_rate = float(countTestError) / float(totalTestPoints)
print("Error rate for the test set: " + str(error_rate))

# Plot the data points
xAxis1 = [i[0] for i in feature1]
yAxis1 = [i[1] for i in feature1]
xAxis2 = [i[0] for i in feature2]
yAxis2 = [i[1] for i in feature2]
xAxis3 = [i[0] for i in feature3]
yAxis3 = [i[1] for i in feature3]
plt.plot(xAxis1, yAxis1, 'r.', xAxis2, yAxis2, 'b^', xAxis3, yAxis3, 'g.')
plt.xlabel('Feature1')
plt.ylabel('Feature2')
plt.title('Feature Plot of all Elements')
plt.show()

# Plot the decision boundaries
plotDecBoundaries(twofeatures, claslabels, samplemeans)
```

```python
"""
Wine Dataset (for question (d & e))
Name: Haolun Cheng
USCID: 1882563827
EE559 HW1
"""

import csv
import sys
import numpy as np
import matplotlib.pyplot as plt
from plotDecBoundaries import plotDecBoundaries

# Two best features selection
min_train_error = sys.maxsize
best_feature_1 = 0
best_feature_2 = 0

for i in range(13):
    for j in range(13):

        clas1xtotal = []
        clas1ytotal = []
        clas2xtotal = []
        clas2ytotal = []
        clas3xtotal = []
        clas3ytotal = []
        allclasstwofeatures = []
        allclasslabels = []
        allclassmeans = []

        # Open train csv file for training the classifier
        with open('wine_train.csv', 'r') as train:
            training_set = csv.reader(train)

            # Train
            for line in training_set:
                x, y, label = line[i], line[j], line[-1]
                allclasstwofeatures.append((float(x), float(y)))
                allclasslabels.append(label)
                if label == '1':
                    clas1xtotal.append(float(x))
                    clas1ytotal.append(float(y))
                elif label == '2':
                    clas2xtotal.append(float(x))
                    clas2ytotal.append(float(y))
                else:
                    clas3xtotal.append(float(x))
                    clas3ytotal.append(float(y))

        train.close()

        # Compute mean for each class
        clas1xmean = np.mean(clas1xtotal)
        clas1ymean = np.mean(clas1ytotal)

        clas2xmean = np.mean(clas2xtotal)
        clas2ymean = np.mean(clas2ytotal)

```

```
        clas3xmean = np.mean(clas3xtotal)
        clas3ymean = np.mean(clas3ytotal)

        clas1_mean_point = np.array((clas1xmean, clas1ymean))
        clas2_mean_point = np.array((clas2xmean, clas2ymean))
        clas3_mean_point = np.array((clas3xmean, clas3ymean))

        # Classify data points (training set)
        countTrainingError = 0
        totalTrainingPoints = 0
        with open('wine_train.csv', 'r') as training:
            train_set = csv.reader(training)

            for line in train_set:
                totalTrainingPoints += 1
                x, y, label = line[i], line[j], line[-1]
                trainPoint = np.array((x, y))
                dist1 = np.linalg.norm(trainPoint.astype(float) -
clas1_mean_point)
                dist2 = np.linalg.norm(trainPoint.astype(float) -
clas2_mean_point)
                dist3 = np.linalg.norm(trainPoint.astype(float) -
clas3_mean_point)

                if dist1 < dist2 and dist1 < dist3:
                    if label != '1':
                        countTrainingError += 1
                if dist2 < dist1 and dist2 < dist3:
                    if label != '2':
                        countTrainingError += 1
                if dist3 < dist1 and dist3 < dist2:
                    if label != '3':
                        countTrainingError += 1

        training.close()

        # Find the two features with minimum errors
        if countTrainingError < min_train_error:
            min_train_error = countTrainingError
            best_feature_1 = i
            best_feature_2 = j

# Fine error rate for training set
error_rate = float(min_train_error) / float(totalTrainingPoints)
print("Error rate for the training set: " + str(error_rate))
print("Best feature 1: " + str(best_feature_1 + 1))
print("Best feature 2: " + str(best_feature_2 + 1))

# Classify three classes and find mean for the best two features
class1 = []
class2 = []
class3 = []
clas1xtotal = []
clas1ytotal = []
clas2xtotal = []
clas2ytotal = []
clas3xtotal = []
clas3ytotal = []
allclasstwofeatures = []
allclasslabels = []
```

```
allclassmeans = []
with open('wine_train.csv') as classify:
    threeclasses = csv.reader(classify)

    for k in threeclasses:
        x, y, label = k[best_feature_1], k[best_feature_2], k[-1]
        allclasstwofeatures.append((float(x), float(y)))
        allclasslabels.append(label)
        if label == '1':
            class1.append((x, y))
            clas1xtotal.append(float(x))
            clas1ytotal.append(float(y))
        elif label == '2':
            class2.append((x, y))
            clas2xtotal.append(float(x))
            clas2ytotal.append(float(y))
        else:
            class3.append((x, y))
            clas3xtotal.append(float(x))
            clas3ytotal.append(float(y))

classify.close()

# Compute mean for each class
clas1xmean = np.mean(clas1xtotal)
clas1ymean = np.mean(clas1ytotal)

clas2xmean = np.mean(clas2xtotal)
clas2ymean = np.mean(clas2ytotal)

clas3xmean = np.mean(clas3xtotal)
clas3ymean = np.mean(clas3ytotal)

clas1_mean_point = np.array((clas1xmean, clas1ymean))
clas2_mean_point = np.array((clas2xmean, clas2ymean))
clas3_mean_point = np.array((clas3xmean, clas3ymean))

# Variables for plotDecBoundaries
allclassmeans = [[clas1xmean, clas1ymean], [clas2xmean, clas2ymean],
[clas3xmean, clas3ymean]]
twofeatures = np.array(allclasstwofeatures).astype(float)
claslabels = np.array(allclasslabels).astype(float)
samplemeans = np.array(allclassmeans).astype(float)

# Plot the data points
xAxis1 = [i[0] for i in class1]
yAxis1 = [i[1] for i in class1]
xAxis2 = [i[0] for i in class2]
yAxis2 = [i[1] for i in class2]
xAxis3 = [i[0] for i in class3]
yAxis3 = [i[1] for i in class3]
plt.plot(xAxis1, yAxis1, 'r.', xAxis2, yAxis2, 'b^', xAxis3, yAxis3, 'g.')
plt.xlabel('Feature1')
plt.ylabel('Feature2')
plt.title('Feature Plot of all Elements')
plt.show()

# Plot the decision boundaries
plotDecBoundaries(twofeatures, claslabels, samplemeans)

```

```
# Classify data points (test set)
countTestError = 0
totalTestPoints = 0
with open('wine_test.csv', 'r') as test:
    test_set = csv.reader(test)

    for line in test_set:
        totalTestPoints += 1
        x, y, label = line[best_feature_1], line[best_feature_2], line[-1]
        testPoint = np.array((x, y))
        dist1 = np.linalg.norm(testPoint.astype(float) - clas1_mean_point)
        dist2 = np.linalg.norm(testPoint.astype(float) - clas2_mean_point)
        dist3 = np.linalg.norm(testPoint.astype(float) - clas3_mean_point)

        if dist1 < dist2 and dist1 < dist3:
            if label != '1':
                countTestError += 1
        if dist2 < dist1 and dist2 < dist3:
            if label != '2':
                countTestError += 1
        if dist3 < dist1 and dist3 < dist2:
            if label != '3':
                countTestError += 1

test.close()

# Fine error rate for test set
error_rate = float(countTestError) / float(totalTestPoints)
print("Error rate for the test set: " + str(error_rate))
```