```
Dec 9, LUL
 1 import matplotlib.pyplot as plt
 2 import numpy as np
 3 import os
   PLOT_COLORS = ['red', 'green', 'blue', 'orange'] # Colors for your plots
                   # Number of Gaussians in the mixture model
 6 K = 4
 7 NUM TRIALS = 3 # Number of trials to run (can be adjusted for debugging)
   UNLABELED = -1 # Cluster label for unlabeled data points (do not change)
                                                       Dec 9, 2021, 12:22:23 PM PST
10
   def main(is semi supervised, trial num):
      """Problem 3: EM for Gaussian Mixture Models (unsupervised and semi-supervised)"""
12
13
       print('Running {} EM algorithm...'
14
              .format('semi-supervised' if is_semi_supervised else 'unsupervised'))
15
16
       # Load dataset
       train_path = os.path.join('.', 'train.csv')
17
       x_all, z_all = load_gmm_dataset(train_path)
18
19
20
       # Split into labeled and unlabeled examples
21
       labeled_idxs = (z_all != UNLABELED).squeeze()
       x_tilde = x_all[labeled_idxs, :] # Labeled examples
22
23
        z_tilde = z_all[labeled_idxs, :] # Corresponding labels
                                         # Unlabeled examples
       x = x_all[~labeled_idxs, :]
24
25
        # *** START CODE HERE ***
26
                                                                          ec 9, 2021, 12:22:23 PN
       # (1) Initialize mu and sigma by splitting the n_examples data points uniformly at random
27
       # into K groups, then calculating the sample mean and covariance for each group
28
       n, d = x.shape
29
        group = np.random.choice(K, n)
30
31
       mu = [np mean(x[group == g, :], axis=0) for g in range(K)]
32
        sigma = [np.cov(x[group == g, :].T) for g in range(K)]
33
34
       # (2) Initialize phi to place equal probability on each Gaussian
35
       # phi should be a numpy array of shape (K,)
36
        phi = np.full((K,), fill_value=(1. / K), dtype=np.float32)
37
38
       # (3) Initialize the w values to place equal probability on each Gaussian
       # w should be a numpy array of shape (m, K)
39
       w = np.full((n, K), fill_value=(1. / K), dtype=np.float32)
40
        # *** END CODE HERE ***
41
42
43
       if is_semi_supervised:
           w = run_semi_supervised_em(x, x_tilde, z_tilde, w, phi, mu, sigma)
44
45
       else:
46
           w = run em(x, w, phi, mu, sigma)
47
                                                   tell@stanford.edu-Dec 9, 2021, 12
       # Plot your predictions
48
49
       z_pred = np.zeros(n)
       if w is not None: # Just a placeholder for the starter code
50
51
           for i in range(n):
52
               z_pred[i] = np.argmax(w[i])
53
54
55
       plot_gmm_preds(x, z_pred, is_semi_supervised, plot_id=trial_num)
56
   def run_em(x, w, phi, mu, sigma):
58
        """Problem 3(d): EM Algorithm (unsupervised).
59
60
        See inline comments for instructions.
61
62
       Args:
           x: Design matrix of shape (n_examples, dim).
63
64
           w: Initial weight matrix of shape (n_examples, k).
           phi: Initial mixture prior, of shape (k,).
65
           mu: Initial cluster means, list of k arrays of shape (dim,).
66
67
            sigma: Initial cluster covariances, list of k arrays of shape (dim, dim).
68
                                                                                              adii-Dec 9
69
        Returns:
           Updated weight matrix of shape (n_examples, k) resulting from EM algorithm.
70
71
           More specifically, w[i, j] should contain the probability of
72
           example x^{(i)} belonging to the j-th Gaussian in the mixture.
```

```
206
        # No need to change any of these parameters
 74
        eps = 1e-3 # Convergence threshold
 75
        max iter = 1000
 76
 77
 78
        # Stop when the absolute change in log-likelihood is < eps
        # See below for explanation of the convergence criterion
 79
        it = 0
 80
        11 = prev_11 = None
                                                                              12:22:23 PM PS1
 81
        while it < max_iter and (prev_ll is None or np.abs(ll - prev_ll) >= eps):
 82
 83
            pass # Just a placeholder for the starter code
 84
            # *** START CODE HERE
 85
            # (1) E-step: Update your estimates in w
 86
            w = e_step(x, w, phi, mu, sigma)
 87
            # (2) M-step: Update the model parameters phi, mu, and sigma
 88
 89
            phi, mu, sigma = m_step(x, w, mu, sigma)
 90
            # (3) Compute the log-likelihood of the data to check for convergence.
 91
            # By log-likelihood, we mean `ll = sum_x[log(sum_z[p(x|z) * p(z)])]`.
 92
            # We define convergence by the first iteration where abs(ll - prev_ll) < eps.
 93
            # Hint: For debugging, recall part (a). We showed that ll should be monotonically increasing.
 94
 95
            prev ll = ll
 96
            ll = log_likelihood(x, phi, mu, sigma)
 97
            it += 1
            print('[iter: {:03d}, log-likelihood: {:.4f}]'.format(it, ll))
 98
                                                                           ec9,2021,12:22:23 PN
            # *** END CODE HERE ***
 99
100
        return w
101
102
103
104
    def run_semi_supervised_em(x, x_tilde, z_tilde, w, phi, mu, sigma):
105
         """Problem 3(e): Semi-Supervised EM Algorithm.
106
107
        See inline comments for instructions.
108
109
        Args:
110
            x: Design matrix of unlabeled examples of shape (n_examples_unobs, dim).
111
            x_tilde: Design matrix of labeled examples of shape (n_examples_obs, dim).
112
            z_tilde: Array of labels of shape (n_examples_obs, 1).
113
            w: Initial weight matrix of shape (n_examples, k).
            phi: Initial mixture prior, of shape (k,).
114
115
            mu: Initial cluster means, list of k arrays of shape (dim,).
116
            sigma: Initial cluster covariances, list of k arrays of shape (dim, dim).
117
118
        Returns:
119
            Updated weight matrix of shape (n_examples, k) resulting from semi-supervised EM algorithm.
                                                                      ford.edu - Dec 9, 2021, 12
120
            More specifically, w[i, j] should contain the probability of
121
            example x^{\wedge}(i) belonging to the j-th Gaussian in the mixture.
122
123
        # No need to change any of these parameters
124
        alpha = 20. # Weight for the labeled examples
125
        eps = 1e-3
                    # Convergence threshold
126
        max_iter = 1000
127
128
        # Stop when the absolute change in log-likelihood is < eps
129
        # See below for explanation of the convergence criterion
130
        it = 0
131
        ll = prev_ll = None
132
        while it < max_iter and (prev_ll is None or np.abs(ll - prev_ll) >= eps):
            pass # Just a placeholder for the starter code
133
            # *** START CODE HERE ***
134
            # (1) E-step: Update your estimates in w
135
            w = e_step(x, w, phi, mu, sigma)
136
137
138
            # (2) M-step: Update the model parameters phi, mu, and sigma
            phi, mu, sigma = m_step_ss(x, x_tilde, z_tilde, w, phi, mu, sigma, alpha)
139
140
141
            # (3) Compute the log-likelihood of the data to check for convergence.
            # Hint: Make sure to include alpha in your calculation of ll.
142
143
            # Hint: For debugging, recall part (a). We showed that ll should be monotonically increasing.
144
            prev_ll = ll
```

H/H/H

73

```
145
           ll = log likelihood(x, phi, mu, sigma)
           ll += alpha * log_likelihood(x_tilde, phi, mu, sigma, z_tilde)
146
147
           it += 1
           print('[iter: {:03d}, log-likelihood: {:.4f}]'.format(it, ll))
148
           # *** END CODE HERE ***
149
150
151
        return w
152
                                                            9,2021,12:22:23 PM PST
153
154
          START CODE HERE
      Helper functions
155
156
157 def e step(x, w, phi, mu, sigma):
        """E-step for both unsupervised and semi-supervised EM."""
158
        n, d = x.shape
159
160
        k = len(mu)
161
162
        for i in range(n):
163
           for j in range(k):
164
               w[i, j] = p_x_given_z(x[i], mu[j], sigma[j]) * phi[j]
                              tanford.edu.
165
166
        w /= np.sum(w, axis=1, keepdims=True)
167
168
        return w
169
170
                                                         7.edu - Dec 9, 2021, 12:22:23 PN
'nus_mu)
171 def m_step(x, w, mu, sigma):
        """M-step for unsupervised EM."""
172
173
        n, d = x.shape
174
        k = len(mu)
175
176
        phi = np.mean(w, axis=0)
177
178
        for j in range(k):
179
           w_{j} = w[:, j:j + 1]
180
            mu[j] = np.sum(w_j * x, axis=0) / np.sum(w_j)
181
182
            sigma[j] = np.zeros like(sigma[j])
183
           for i in range(n):
184
               x_{minus_mu} = x[i] - mu[j]
185
               sigma[j] += w[i, j] * np.outer(x_minus_mu, x minus_mu)
                                 atel1@stani
186
            sigma[j] /= np.sum(w_j)
187
188
        return phi, mu, sigma
189
190
191 def m_step_ss(x, x_tilde, z_tilde, w, phi, mu, sigma, alpha):
          192
        """M-step for semi-supervised EM."""
193
        n, _ = x.shape
194
        n_tilde, __ = x_tilde.shape
195
        k = len(mu)
196
197
        w_{colsums} = np.sum(w, axis=0)
198
        k_counts = [np.sum(z_tilde == j) for j in range(k)]
199
        for j in range(k):
200
201
202
203
204
205
206
207
            sigma[j] = np.zeros_like(sigma[j])
           for i in range(n):
208
209
               x_{minus_mu} = x[i] - mu[j]
210
               sigma[j] += w[i, j] * np.outer(x_minus_mu, x_minus_mu)
211
           for i in range(n_tilde):
212
               if z_tilde[i] == j:
213
                   x_{minus_mu} = x_{tilde[i]} - mu[j]
214
                   sigma[j] += alpha * np.outer(x_minus_mu, x_minus_mu)
215
            sigma[j] /= (np.sum(w_j) + alpha * k_counts[j])
216
```

adu-Dec 9

```
ec 9, 202
218
219
220 def log_likelihood(x, phi, mu, sigma, z=None):
        """Get log-likelihood of the data `x` given model parameters
221
         `phi`, `mu`, and `sigma`.
222
223
                                        rd.edu - Dec 9, 2021, 12:22:23 PM PST

vole 'x'

vole 'x'
224
        n, d = x.shape
225
        k = len(phi)
        11 = 0.
226
227
        for i in range(n):
228
            if z is None: # Unsupervised case
229
                p_x = 0.
230
                for j in range(k):
231
                    p_x += p_x_given_z(x[i], mu[j], sigma[j]) * phi[j]
232
            else: # Supervised case
233
                j = int(z[i])
                p_x = p_x_given_z(x[i], mu[j], sigma[j]) * phi[j]
234
235
            ll += np.log(p_x)
236
237
        return 11
238
239
240
    def p_x_given_z(x, mu, sigma):
241
        """Get probability of a single example `x` given model parameters
242
         'mu' and 'sigma' (corresponding to cluster z = j).
                                                            J.edu - Dec 9, 2021, 12:22:23 PN
243
        d = len(x)
244
245
        assert d == len(mu) and sigma.shape == (d, d), 'Shape mismatch.'
246
247
        c = 1. / ((2. * np.pi) ** (d / 2) * np.sqrt(np.linalg.det(sigma)))
248
        x minus mu = x - mu
        sigma_inv = np.linalg.inv(sigma)
249
250
        p_val = c * np.exp(-.5 * x_minus_mu.dot(sigma_inv).dot(x_minus_mu.T))
251
252
        return p_val
      *** END CODE HERE ***
253
254
255
256 def plot_gmm_preds(x, z, with_supervision, plot_id):
        """Plot GMM predictions on a 2D dataset `x` with labels
257
258
        Write to the output directory, including `plot id`
259
        in the name, and appending 'ss' if the GMM had supervision.
260
261
262
        NOTE: You do not need to edit this function.
263
                                                                             edu-Dec 9, 2021, 12
264
        plt.figure(figsize=(12, 8))
265
        plt.title('{} GMM Predictions'.format('Semi-supervised' if with_supervision else 'Unsupervised'))
266
        plt.xlabel('x 1')
267
        plt.ylabel('x_2')
268
269
        for x_1, x_2, z_i in zip(x[:, 0], x[:, 1], z):
270
            color = 'gray' if z_ < 0 else PLOT_COLORS[int(z_)]</pre>
271
            alpha = 0.25 if z < 0 else 0.75
272
            plt.scatter(x_1, x_2, marker='.', c=color, alpha=alpha)
273
                                                 oatel1@stanh
        file_name = 'pred{}_{}.pdf'.format('_ss' if with_supervision else '', plot_id)
274
275
        save_path = os.path.join('.', file_name)
276
        plt.savefig(save_path)
277
278
279 def load gmm dataset(csv path):
        """Load dataset for Gaussian Mixture Model
280
281
282
        Args:
             csv_path: Path to CSV file containing dataset.
283
284
                                                                                               adu-Dec 9
285
        Returns:
286
            x: NumPy array shape (n_examples, dim)
287
            z: NumPy array shape (n_exampls, 1)
288
```

217

return phi, mu, sigma

```
NOTE: You do not need to edit this function.
289
                                       Dec 9,
290
291
292
        # Load headers
293
        with open(csv_path, 'r') as csv_fh:
294
            headers = csv_fh.readline().strip().split(',')
295
296
        # Load features and labels
                                              du-Dec 9, 2021, 12:22:23 PM PST
        x cols = [i for i in range(len(headers)) if headers[i].startswith('x')]
297
298
        z_cols = [i for i in range(len(headers)) if headers[i] == 'z']
299
300
       x = np.loadtxt(csv_path, delimiter=',', skiprows=1, usecols=x_cols, dtype=float)
301
        z = np.loadtxt(csv_path, delimiter=',', skiprows=1, usecols=z_cols, dtype=float)
302
303
        if z.ndim == 1:
304
            z = np.expand_dims(z, axis=-1)
305
306
        return x, z
307
308
309 if
       name == '__main__':
310
        np.random.seed(229)
311
        # Run NUM TRIALS trials to see how different initializations
312
        # affect the final predictions with and without supervision
313
        for t in range(NUM_TRIALS):
                        vgPatel1@stanford.edu - Dec 9, 2021, 12:22:23 PN
314
           main(is_semi_supervised=False, trial_num=t)
315
            # *** START CODE HERE ***
316
317
            # Once you've implemented the semi-supervised version,
           # uncomment the following line.
318
            # You do not need to add any other lines in this code block.
319
320
321
           # main(is_semi_supervised=True, trial_num=t)
322
323
            # *** END CODE HERE ***
```



adii-Dec9

12:22:23 PM PST