CSE 150 Homework 5

Pedro Sousa Meireles
A15677282

Fall 2018

1 Maximum Likelihood Estimation

(a) **Complete data**

$$P(B=b|A=a) = \frac{\sum_{t=1}^{T} I(a,a_t)\cdot I(b,b_t)}{\sum_{t=1}^{T} I(a,a_t)}$$

$$P(C=c|A=a,B=b) = \frac{\sum_{t=1}^{T} I(a,a_t)\cdot I(b,b_t)\cdot I(c,c_t)}{\sum_{t=1}^{T} I(a,a_t)\cdot I(b,b_t)}$$

$$P(D=d|A=a,C=c) = \frac{\sum_{t=1}^{T} I(a,a_t)\cdot I(c,c_t)\cdot I(d,d_t)}{\sum_{t=1}^{T} I(a,a_t)\cdot I(d,d_t)}$$

(b) **Posterior probability**

$$\begin{aligned} P(a,c|b,d) &= \frac{P(a,b,c,d)}{P(b,d)} \\ &= \frac{P(a)\cdot P(b|a)\cdot P(c|a,b)\cdot P(d|a,c)}{\sum_{a'}\sum_{c'} P(a=a')\cdot P(b|a=a')\cdot P(c=c'|a=a',b)\cdot P(d|a=a',c=c')} \end{aligned}$$

(c) **Posterior probability**

$$P(a|b,d) = \sum_{c'} P(a,c=c'|b,d)$$

$$P(c|b,d) = \sum_{a'} P(a=a',c|b,d)$$

(d) **Log-likelihood**

$$\begin{aligned} L &= \sum_t \log P(B=b_T,D=d_t) \\ &= \sum_t \log \sum_a \sum_c P(A=a,B=b_T,C=c,D=d_t) \\ &= \sum_t \log \sum_a \sum_c P(A=a)\cdot P(B=b_T|A=a)\cdot P(C=c|A=a,B=b_t)\cdot P(D=d_t|A=a,C=c) \end{aligned}$$

(e) **EM algorithm**

$$\begin{aligned} P(A=a) &\longleftarrow \frac{\widehat{count}(A=a)}{T} \\ &\longleftarrow \frac{\sum_{t=1}^{T} P(A=a|B=b_t,D=d_t)}{T} \\ &\longleftarrow \frac{\sum_{t=1}^{T}\sum_c P(A=a,C=c|B=b_t,D=d_t)}{T} \end{aligned}$$

$$
\begin{aligned}
P(B=b|A=a) &\longleftarrow \frac{\widehat{count}(A=a,B=b)}{\widehat{count}(A=a)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} P(A=a,B=b|B=b_t,D=d_t)}{\widehat{count}(A=a)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} I(b,b_t)\cdot P(A=a|B=b_t,D=d_t)}{\widehat{count}(A=a)} \\
&\longleftarrow \frac{\sum_{t=1}^{T}\sum_c I(b,b_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\widehat{count}(A=a)} \\
&\longleftarrow \frac{\sum_{t=1}^{T}\sum_c I(b,b_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\sum_{t=1}^{T}\sum_c P(A=a,C=c|B=b_t,D=d_t)}
\end{aligned}
$$

$$
\begin{aligned}
P(C=c|A=a,B=b) &\longleftarrow \frac{\widehat{count}(A=a,B=b,C=c)}{\widehat{count}(A=a,B=b)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} P(A=a,B=b,C=c|B=b_t,D=d_t)}{\widehat{count}(A=a,B=b)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} I(b,b_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\widehat{count}(A=a,B=b)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} I(b,b_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\sum_{t=1}^{T}\sum_{c'} I(b,b_t)\cdot P(A=a,C=c'|B=b_t,D=d_t)}
\end{aligned}
$$

$$
\begin{aligned}
P(D=d|A=a,C=c) &\longleftarrow \frac{\widehat{count}(A=a,C=c,D=d)}{\widehat{count}(A=a,C=c)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} P(A=a,C=c,D=d|B=b_t,D=d_t)}{\widehat{count}(A=a,C=c)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} I(d,d_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\widehat{count}(A=a,C=c)} \\
&\longleftarrow \frac{\sum_{t=1}^{T} I(d,d_t)\cdot P(A=a,C=c|B=b_t,D=d_t)}{\sum_{t=1}^{T} P(A=a,C=c|B=b_t,D=d_t)}
\end{aligned}
$$

2 EM algorithm for noisy-OR

(a) **Equivalence of models**

- (i) Marginalization over $\overrightarrow{z}$
- (ii) Product/chain rule was applied along with conditional independence between every z_i since all paths between different z_i are d-separated due to rule #3.
- (iii) $P_B(Y=0|\overrightarrow{Z}=\overrightarrow{z})=0$ for any $z_i=1$, so the only remaining term is for $\overrightarrow{Z}=0$.
- (iv) $P_B(Z_i=0|X_i=1)=1-p_i$ and $P_B(Z_i=0|X_i=0)=1$. Then $P_B(Z_i=0|X_i=x_i)=(1-p_i)^{x_i}$

(b) **EM Implementation: Per-iteration statistics**

iteration	number of mistakes M	log conditional likelihood L
0	175	-0.9581
1	56	-0.4959
2	43	-0.4082
4	42	-0.3646
8	44	-0.3475
16	40	-0.1146
32	37	-0.3226
64	37	-0.3148
128	36	-0.3112
256	36	-0.3102
512	36	-0.3100

(c) **EM Implementation: Estimated values for p_i**

i	p_i
1	7.952606636558421e-05
2	0.00481741209369718
3	2.5702471621528087e-11
4	0.26533320243361963
5	1.4919335171144578e-05
6	0.009464229795569459
7	0.24030074435506368
8	0.11345165109907844
9	0.0001434658634790401
10	0.5234804065754611
11	0.4072853322527083
12	9.074568784373926e-08
13	0.6157947609276805
14	5.954608022848239e-06
15	0.04490171238688918
16	0.5899773388133327
17	0.9999999999999938
18	0.9999999821990931
19	4.154500914170821e-09
20	0.4629914874582514
21	0.3531984001127537
22	0.5248644134908856
23	0.19475859952835034

(d) **EM Implementation: Source code**

```
In [99]: import math
```

```
In [100]: X = []
          for l in open("hw5_noisyOr_x.txt"):
              numbers = l.split()
              numbers = [int(s) for s in numbers]
              X.append(numbers)
```

```
In [101]: y = []
          for l in open("hw5_noisyOr_y.txt"):
              #numbers = l.split()
              #numbers = [int(s) for s in numbers]
              y.append(int(l))
```

```
In [102]: p = []
          for i in range(0, 23):
              p.append(0.05)

In [103]: def updatePi():
              newP = []
              for i in range(0, 23):
                  pi = 0
                  Ti = 0
                  for t in range(0, len(X)):
                      prod = 1
                      for j in range(0, len(X[0])):
                          prod *= (1-p[j])**X[t][j]
                      pi += y[t]*X[t][i]*p[i]/(1-prod)
                      Ti += X[t][i]
                  pi = pi/Ti
                  newP.append(pi)
              return newP

In [104]: def computePYX():
              PYX = []
              for i in range(0,len(X)):
                  pyx = 1
                  for j in range(0, len(X[0])):
                      pyx *= (1-p[j])**X[i][j]
                  PYX.append(1-pyx)
              return PYX

In [105]: def loglikelihood(probabilities):
              ll = 0
              for i in range(len(y)):
                  if y[i] == 0:
                      ll += math.log(1-probabilities[i])
                  else:
                      ll += math.log(probabilities[i])
              ll /= len(y)
              return ll

In [106]: iterations = []

          probabilities = computePYX()
          ll = loglikelihood(probabilities)

          errors = 0
          for i in range(0, len(y)):
              if (probabilities[i] >= 0.5 and y[i] == 0)
              or (probabilities[i] < 0.5 and y[i] == 1):
                  errors +=1

          iterations.append((errors, ll))

          for i in range(0,512):
              p = updatePi();
              probabilities = computePYX()
              ll = loglikelihood(probabilities)
```

```
    errors = 0
    for i in range(0, len(y)):
        if (probabilities[i] >= 0.5 and y[i] == 0)
        or (probabilities[i] < 0.5 and y[i] == 1):
            errors +=1

    iterations.append((errors, ll))
```

In [120]:
```
table = []
table.append((0, iterations[0][0], iterations[0][1]))

i = 1
while i <= 512:
    table.append((i, iterations[i][0], iterations[i][1]))
    i *= 2
```

In [121]:
```
table
```

Out[121]:
```
[(0, 175, -0.9580854082157914),
 (1, 56, -0.49591639407753635),
 (2, 43, -0.40822081705839114),
 (4, 42, -0.3646149825001877),
 (8, 44, -0.34750061620878253),
 (16, 40, -0.33461704895854844),
 (32, 37, -0.3225814031674978),
 (64, 37, -0.3148266983628557),
 (128, 36, -0.3111558472151897),
 (256, 36, -0.3101613534740759),
 (512, 36, -0.30999030298497576)]
```

In [122]:
```
p
```

Out[122]:
```
[7.952606636558421e-05,
 0.004817412093697185,
 2.5702471621528087e-11,
 0.26533320243361963,
 1.4919335171144578e-05,
 0.009464229795569459,
 0.24030074435506368,
 0.11345165109907844,
 0.0001434658634790401,
 0.5234804065754611,
 0.4072853322527083,
 9.074568784373926e-08,
 0.6157947609276805,
 5.954608022848239e-06,
 0.04490171238688918,
 0.5899773388133327,
 0.9999999999999938,
 0.9999999821990931,
 4.154500914170821e-09,
 0.4629914874582514,
 0.3531984001127537,
 0.5248644134908856,
 0.19475859952835034]
```

3 EM algorithm for binary matrix completion

(a) **Sanity check**

(b) **Likelihood**

(c) **E-step**

(d) **M-step**

(e) **Implementation**

(f) **Personal categorization**

(g) **Personal movie recommendations**

(h) **Source code**