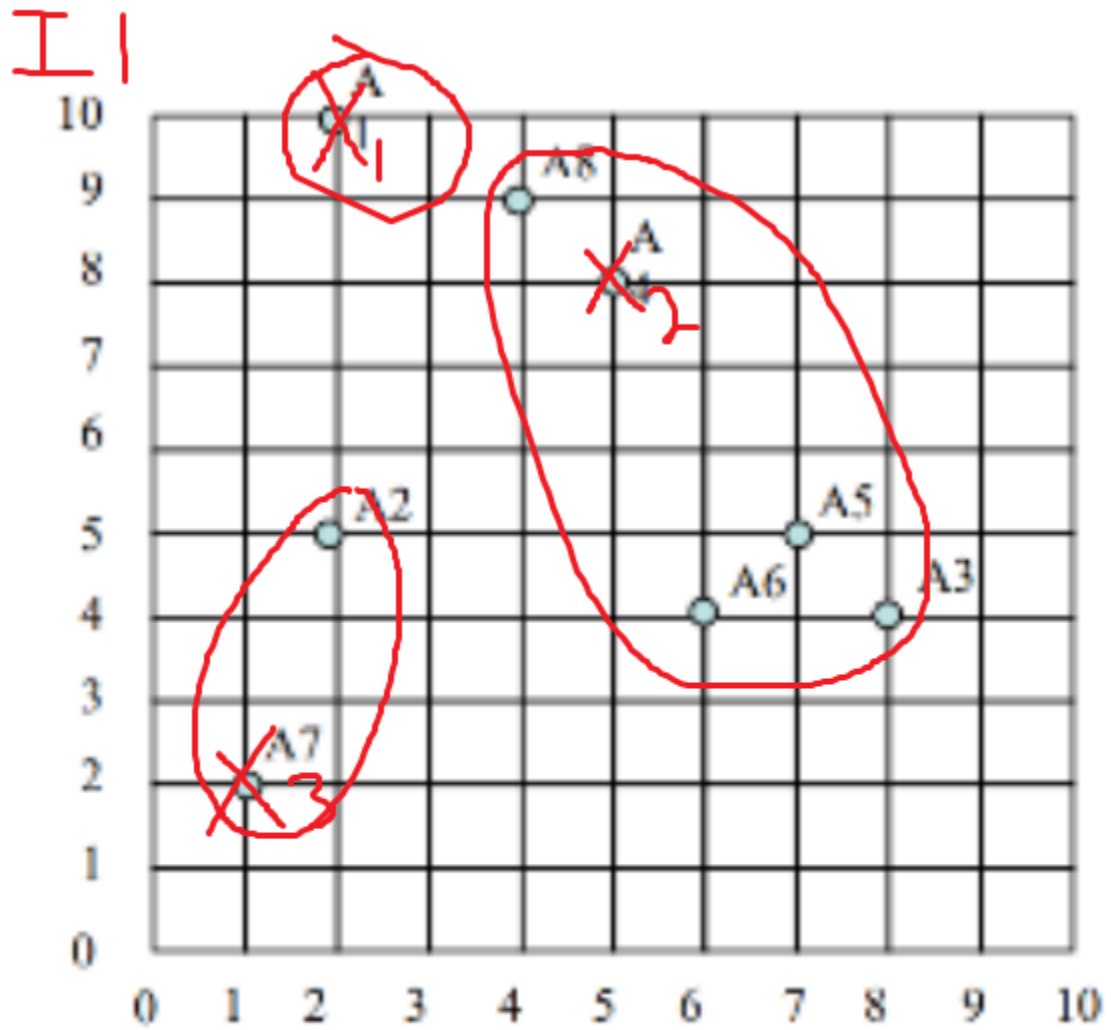1. a)
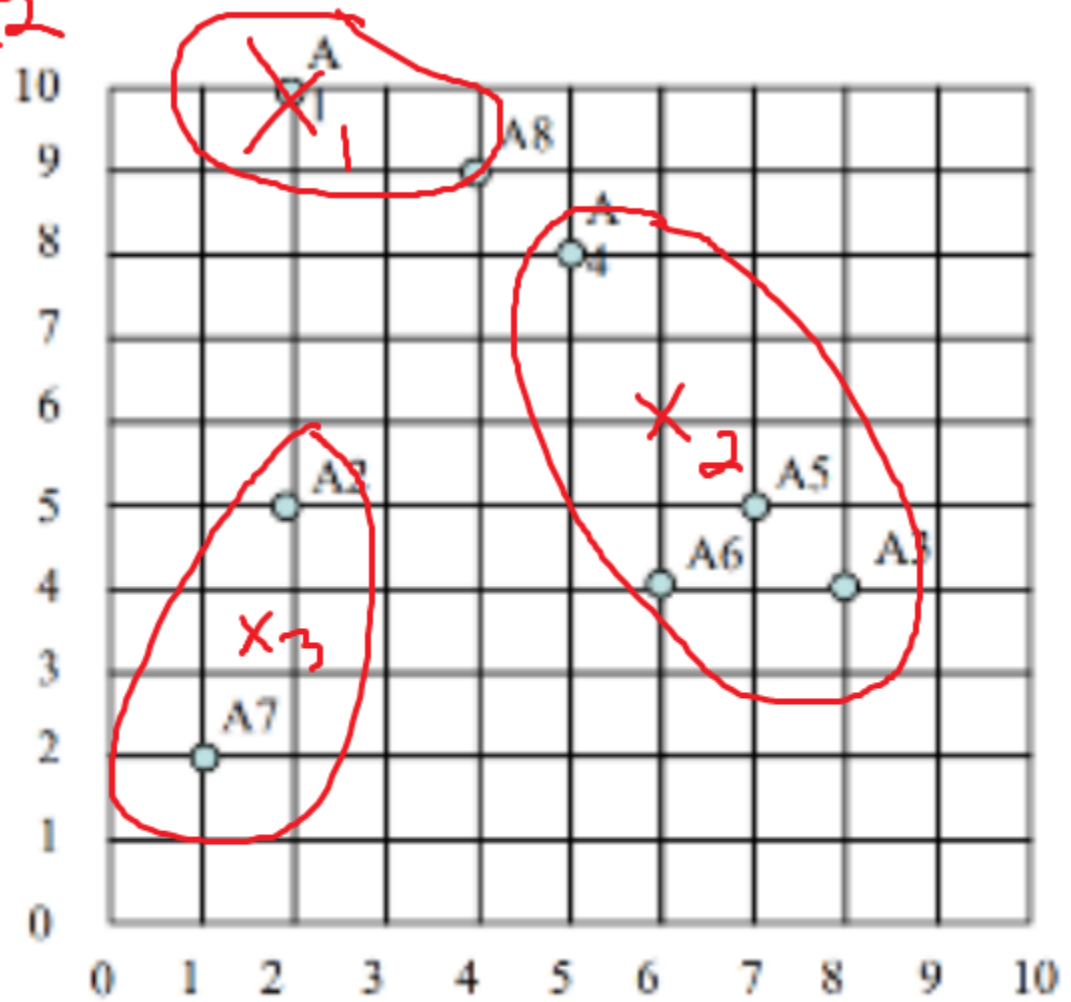

Mean1 = (2, 10)/1 = (2, 10)

Mean2 = (4+5+6+7+8)/5, (9+8+5+4+4)/5 = (6, 6)

Mean3 = (1+2)/2, (2+5)/2 = (1.5, 3.5)

Move centroids to means. (movement will be shown
at start of next iteration for similicpity of plot.

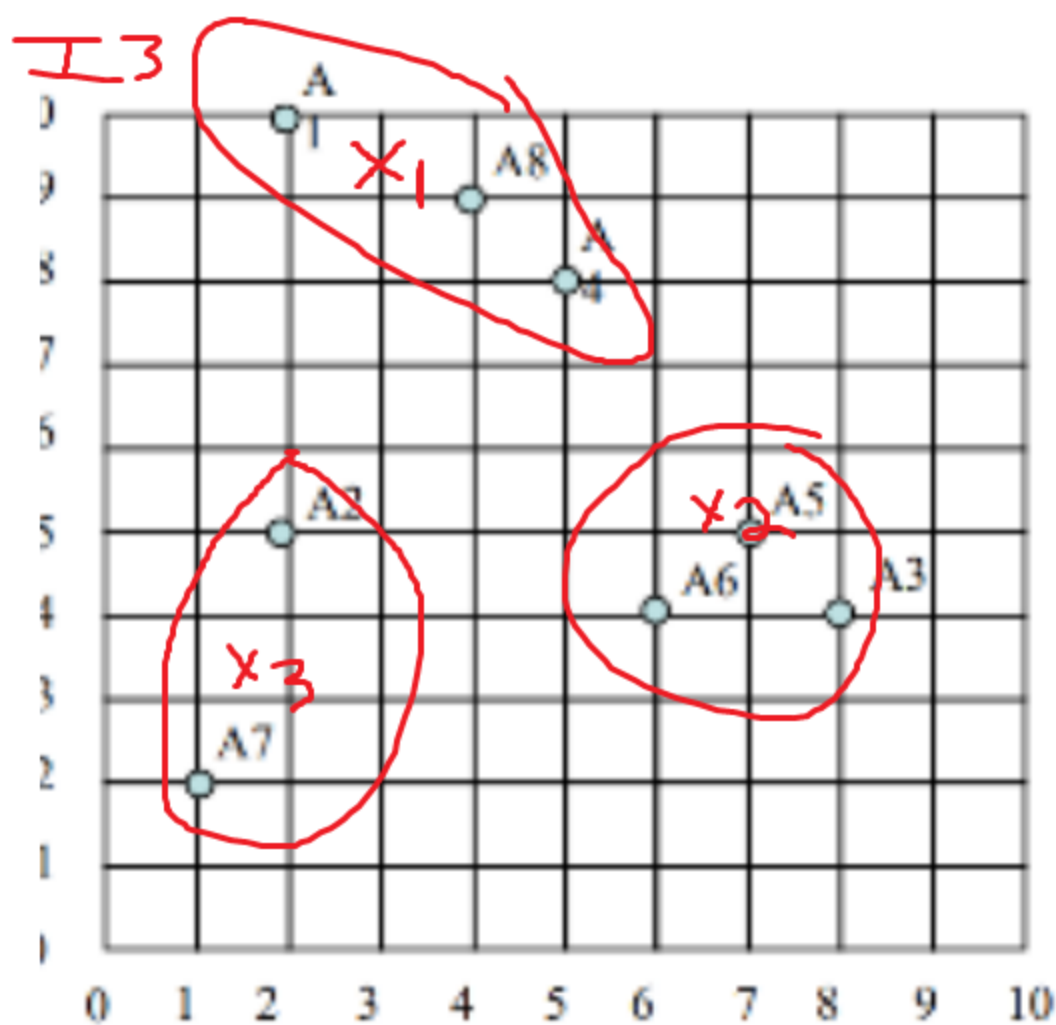Mean1 = (2+4)/2, (9+10)/2 = (3, 9.5)

Mean2 = (5+6+7+8)/4, (4+4+5+8)/4 = (6.5, 5.25)

Mean3 = (1+2)/2, (2+5)/2 = (1.5, 3.5)
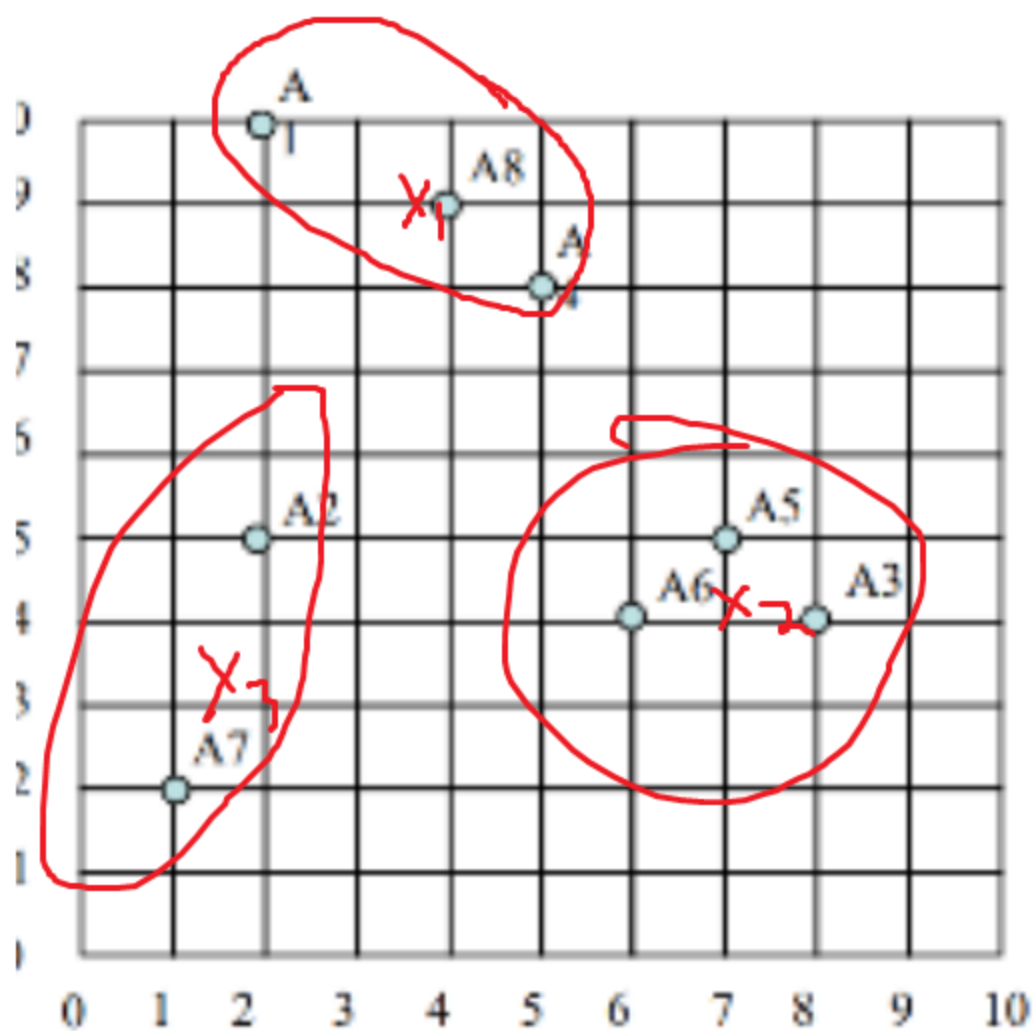
Mean1 = (2+4+5)/3, (8+9+10)/3 = (3.67, 9)

Mean2 = (6+7+8)/3, (4+4+5)/3 = (7, 4.33)

Mean3 = (1.5,3.5), unchanged

Mean1 = (3.67, 9), unchanged
Mean2 = (7,4.33), unchanged
Mean3 = (1.5,3.5), unchanged
Terminate K means algorithm.

```
iteration   =  1
Centroid    =  C1 = (2, 10), C2 = (5, 8), C3 = (1, 2)
instances   =  ['   A1  ', '   A2  ', '   A3  ', '   A4  ', '   A5  ', '   A6  ', '   A7  ', '   A8  ']
C1 distances =  ['   0.00', '   5.00', '   8.49', '   3.61', '   7.07', '   7.21', '   8.06', '   2.24']
C2 distances =  ['   3.61', '   4.24', '   5.00', '   0.00', '   3.61', '   4.12', '   7.21', '   1.41']
C3 distances =  ['   8.06', '   3.16', '   7.28', '   7.21', '   6.71', '   5.39', '   0.00', '   7.62']
iteration 2 centroids = C1 = (2.0, 10.0), C2 = (6.0, 6.0), C3 = (1.5, 3.5)

iteration   =  2
Centroid    =  C1 = (2.0, 10.0), C2 = (6.0, 6.0), C3 = (1.5, 3.5)
instances   =  ['   A1  ', '   A2  ', '   A3  ', '   A4  ', '   A5  ', '   A6  ', '   A7  ', '   A8  ']
C1 distances =  ['   0.00', '   5.00', '   8.49', '   3.61', '   7.07', '   7.21', '   8.06', '   2.24']
C2 distances =  ['   5.66', '   4.12', '   2.83', '   2.24', '   1.41', '   2.00', '   6.40', '   3.61']
C3 distances =  ['   6.52', '   1.58', '   6.52', '   5.70', '   5.70', '   4.53', '   1.58', '   6.04']
iteration 3 centroids = C1 = (3.0, 9.5), C2 = (6.5, 5.25), C3 = (1.5, 3.5)

iteration   =  3
Centroid    =  C1 = (3.0, 9.5), C2 = (6.5, 5.25), C3 = (1.5, 3.5)
instances   =  ['   A1  ', '   A2  ', '   A3  ', '   A4  ', '   A5  ', '   A6  ', '   A7  ', '   A8  ']
C1 distances =  ['   1.12', '   4.61', '   7.43', '   2.50', '   6.02', '   6.26', '   7.76', '   1.12']
C2 distances =  ['   6.54', '   4.51', '   1.95', '   3.13', '   0.56', '   1.35', '   6.39', '   4.51']
C3 distances =  ['   6.52', '   1.58', '   6.52', '   5.70', '   5.70', '   4.53', '   1.58', '   6.04']
iteration 4 centroids = C1 = (3.6666666666666665, 9.0), C2 = (7.0, 4.333333333333333), C3 = (1.5, 3.5)

iteration   =  4
Centroid    =  C1 = (3.6666666666666665, 9.0), C2 = (7.0, 4.333333333333333), C3 = (1.5, 3.5)
instances   =  ['   A1  ', '   A2  ', '   A3  ', '   A4  ', '   A5  ', '   A6  ', '   A7  ', '   A8  ']
C1 distances =  ['   1.94', '   4.33', '   6.62', '   1.67', '   5.21', '   5.52', '   7.49', '   0.33']
C2 distances =  ['   7.56', '   5.04', '   1.05', '   4.18', '   0.67', '   1.05', '   6.44', '   5.55']
C3 distances =  ['   6.52', '   1.58', '   6.52', '   5.70', '   5.70', '   4.53', '   1.58', '   6.04']
iteration 5 centroids = C1 = (3.6666666666666665, 9.0), C2 = (7.0, 4.333333333333333), C3 = (1.5, 3.5)
```
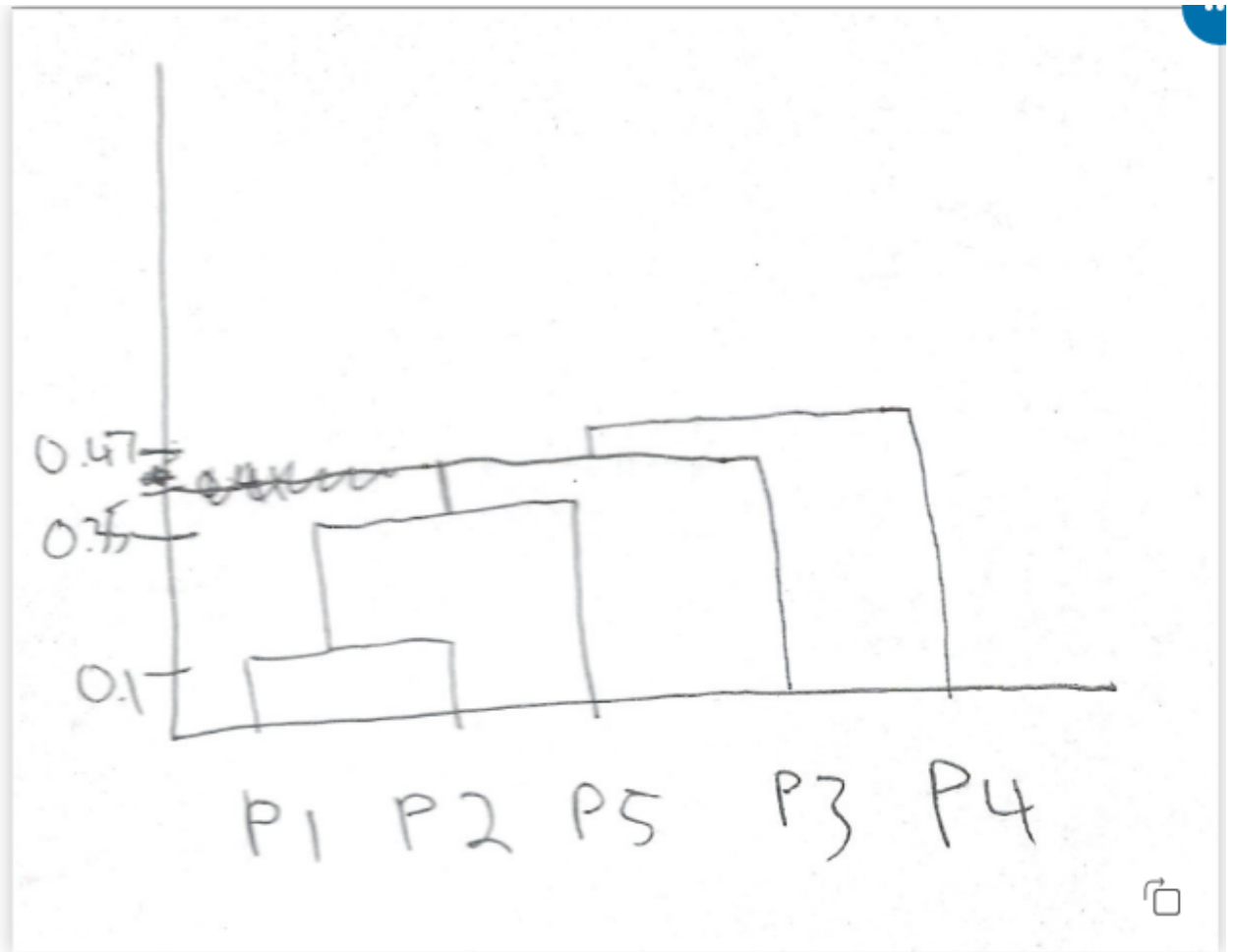
b)   SSE of cluster1 = 6.67
        SSE of cluster2 = 2.67
        SSE of cluster3 = 5
        SSE of final clustering = 14.33

2. a)

The y-axis shows values 0.47, 0.35, 0.1. The x-axis is labeled P1 P2 P5 P3 P4 (a dendrogram).

b)   k = 2, cluster1 = {p1, p2, p5, p3}, cluster2 = {p4}
     k =3, cluster1 = {p1,p2,p5}, cluster2 = {p3}, cluster3 = {p4}
     k = 4, cluster1 = {p1, p2}, cluster2 = {p5}, cluster3={p3}, cluster4={p4}

3. https://github.com/garyjsk271/Machine-Learning.git

4. a)
   cos(John, Fred) = 0.85
   cos(John,Lilian) = 0.99
   cos(John, Cathy) = 0.95

   Compute weighted combination of Lilian, Cathy's ratings.
   For kiss:
   2.25 + (0.99*(3 - 2.125) + 0.95*(1.5 - 2.25)) / (0.99 + 0.95) = 2.33

   For Guns n Roses:
   2.25 + (0.99*(1 - 2.125) + 0.95*(2 - 2.25)) / (0.99 + 0.95) = 1.55

   For Kiss, weighted combination of ratings > 2.0, recommend to John.
   For Guns n' Roses, weighted combination of ratings < 2.0, do not recommend to John.

   b)
   For Kiss, find all cosine similarities.
   cos(Kiss, BonJovi) = 0.99
   cos(Kiss, Metallica) = 0.77
   cos(Kiss, Scorpians) = 0.94
   cos(Kiss, ACDC) = 0.82

   Use BonJovi and Scorpians to compute rating.
   1.83 + (0.99*(3 - 2) + 0.94*(2 - 1.83)) / (0.99 + 0.94) = 2.43

   Since rating is > 2.0, recommend Kiss to John.

   For Guns n Roses, find all cosine similarities.
   cos(Guns n Roses, BonJovi) = 0.71
   cos(Guns n Roses, Metallica) = 0.99
   cos(Guns n Roses, Scorpians) = 0.88
   cos(Guns n Roses, ACDC) = 0.97

   Use Metallica, ACDC to compute rating.
   2 + (0.99*(2 - 2.17) + 0.97*(2 - 2.67)) / (0.99 + 0.97) = 1.59

   Since rating is < 2.0, do not recommend Guns n Roses to John.

5. a)

|T| = 10

Support(a) = 5/10 = 0.5
Support(b) = 7/10 = 0.7
Support(c) = 5/10 = 0.5
Support(d) = 8/10 = 0.8
Support(e) = 6/10 = 0.6

no pruning as all supports > 0.3

F1 = { {a}, {b}, {c}, {d}, {e} }

Support(ab) = 3/10 = 0.3
Support(ac) = 2/10 = 0.2   ***prune**
Support(ad) = 4/10 = 0.4
Support(ae) = 4/10 = 0.4
Support(bc) = 2/10 = 0.2   ***prune**
Support(bd) = 5/10 = 0.5
Support(be) = 4/10 = 0.4
Support(cd) = 4/10 = 0.4
Support(ce) = 2/10 = 0.2   ***prune**
Support(de) = 6/10 = 0.6

F2 ={ {ab}, {ad}, {ae}, {bd}, {be}, {cd}, {de} }

F2 x F2 after merge = { {abd}, {abe}, {bde}, {ade} }
Support(abd) = 2/10 = 0.2   ***prune**
Support(abe) = 2/10 = 0.2   ***prune**
Support(bde) = 4/10 = 0.4
Support(ade) = 4/10 = 0.4

F3 = { {bde}, {ade} }

Rank by all frequent items by support.
d = 0.8
b = 0.7
e, de = 0.6
a, c, bd = 0.5
ad, ae, be, cd, bde, ade = 0.4
ab = 0.3

b)   F3 = { {bde}, {ade} }

Support(bde) = 4/10 = 0.4
Support(ade) = 4/10 = 0.4

b → de
Support(b) = 7/10 = 0.7
satisfies the support requirement.
confidence = Support(bde) / Support(b) = 4/7 = 0.57
does not satisfy the confidence requirement.

d → be
Support(d) = 8/10 = 0.8
satisfies the support requirement.
confidence = Support(bde) / Support(d) = 4/8 = 0.50
does not satisfy the confidence requirement.

e → bd
Support(e) = 6/10 = 0.6
satisfies the support requirement.
confidence = Support(bde) / Support(e) = 4/6 = 0.67
satisfies the confidence requirement.

bd → e
Support(bd) = 5/10 = 0.5
satisfies the support requirement.
confidence = Support(bde) / Support(bd) = 4/5 = 0.8
satisfies the confidence requirement.

be → d
Support(be) = 4/10 = 0.4
satisfies the support requirement.
confidence = Support(bde) / Support(be) = 4/4 = 1.0
satisfies the confidence requirement.

de → b
Support(de) = 6/10 = 0.6
satisfies the support requirement.
confidence = Support(bde) / Support(de) = 4/6 = 0.67
satisfies the confidence requirement.

a → de
Support(a) = 5/10 = 0.5
Satisfies support requirement.
confidence = Support(ade) / Support(a) = 4/5 = 0.8
satisfies the confidence requirement.

d → ae
Support(d) = 8/10 = 0.8
Satisfies support requirement.
confidence = Support(ade) / Support(a) = 4/8 = 0.50
satisfies the confidence requirement.

e → ad
Support(e) = 6/10 = 0.6
Satisfies support requirement.
confidence = Support(ade) / Support(e) = 4/6 = 0.67
satisfies the confidence requirement.

ad → e
Support(ad) = 4/10 = 0.4
Satisfies support requirement.
confidence = Support(ade) / Support(ad) = 4/4 = 1.0
satisfies the confidence requirement.

ae → d
Support(ae) = 4/10 = 0.4
Satisfies support requirement.
confidence = Support(ade) / Support(ae) = 4/4 = 1.0
satisfies the confidence requirement.

de → a
Support(de) = 6/10 = 0.6
Satisfies support requirement.
confidence = Support(ade) / Support(de) = 4/6 = 0.67
satisfies the confidence requirement.

c)   F2 = {ab}, {ad}, {ae}, {bd}, {be}, {cd}, {de}

F2 x F2 = {
       (ab, ad), (ab, ae), (ab, bd), (ab, be), (ab, cd), (ab, de),
       (ad, ae), (ad, bd), (ad,be), (ad,cd), (ad, de),
       (ae, bd), (ae, be), (ae, cd), (ae, de),
       (bd, be), (bd, cd), (bd, de),
       (be, cd), (be, de),
       (cd, de)
}

Merge all pairs with matching prefix size of 1.
C3 = {abd, abe, bde, ade}

Support(abd) = 2/10 = 0.2   *prune
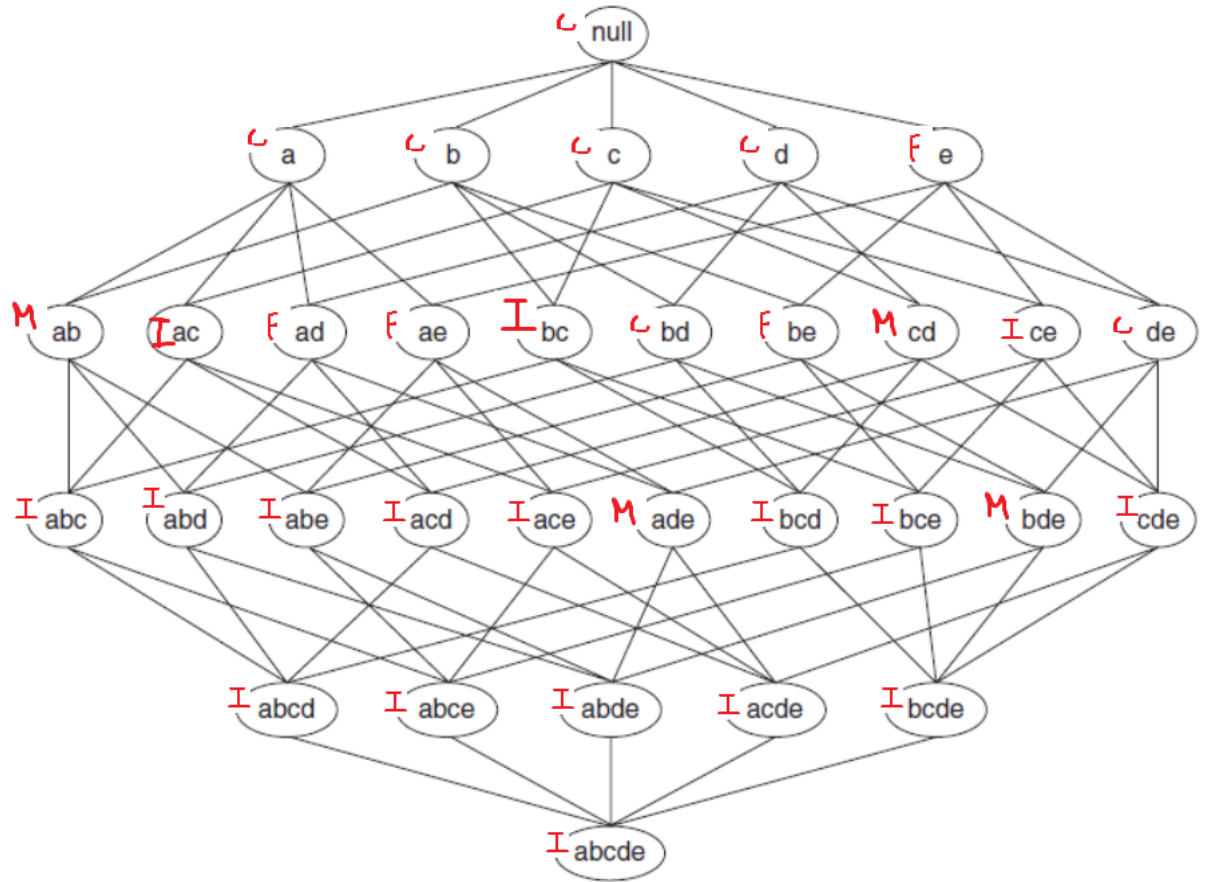Support(abe) = 3/10 = 0.2   *prune
Support(bde) = 4/10 = 0.4
Support(ade) = 4/10 = 0.4

F3 = { ade, bde }

d)

null

a b c d e

M ab   I ac   F ad   F ae   I bc   bd   F be   M cd   I ce   de

I abc   I abd   I abe   I acd   I ace   M ade   I bcd   I bce   M bde   I cde

I abcd   I abce   I abde   I acde   I bcde

I abcde

6. https://github.com/garyjsk271/Machine-Learning.git