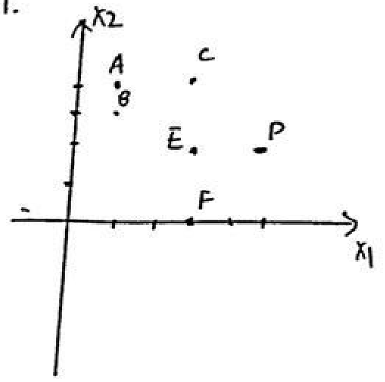
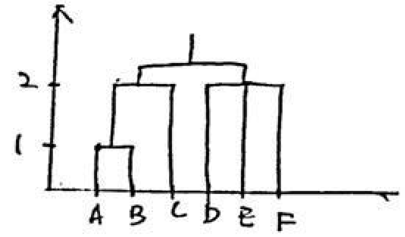
5241 HW5
Haiqi Li
hl3115

Problem1.





step 1 {A,B}
step 2 {E, D, F} {A, B, C} There exists multiple way to achieve.
step 3 {ABCDEF} This is only one of them.

Problem2. $c \in C = \{spam, email\}$ x is sample

$$p(c|x) = \frac{p(x|c)p(c)}{p(x)} \propto p(x|c)p(c) = p(w_1 \cdots w_M|c)\,p(c) = \prod_{i=3}^{M} p(w_i|w_{i-2}, w_{i-1}, c)$$ (Last equation is assumption of 3-gram)

Let $t_k = w_i|w_{i-2}, w_{i-1}$

Let $\mathbb{D}$ be ~~dictionary~~ document space, the $\mathbb{D} = X_s + X_e$

V be vocabulary space, i.e. V = vocabulary($\mathbb{D}$) define V^3 be combination of $(v_i|v_j,v_k)$ v_i is $\forall$ word in V. So element(V^3) = (element(V))^3 $k \in \{i : i \in N$ and $i \leq$ element(V^3)$\}$.

prior $\hat{p}(c) = \frac{N_c}{N}$ $N = \#(X_s + X_e)$ $N_c = \#(X_c)$ $c \in C = \{spam, email\}$.

$$\hat{p}(t|c) = \frac{T_{ct}}{\sum_{t' \in V^3} T_{ct'}}$$ T_{ct} is # of t appear in X_c.

Note use add-one smooth to avoid 0-probability $\hat{p}(t_k|c) = \frac{T_{ct_k}+1}{\sum_{t' \in V^3}(T_{ct'}+1)}$

· We are in a multinomial (3 element) distribution.

x is ~~test~~ sample. $(w_1 \cdots w_M)$

```
. train (C, D) {
  V^3 <- (Vocabulary(D))^3
  N <- #(D)
  for c ∈ C
    prior[c] <- Nc/N
    tc <- get word of combination (D, c)
    for t ∈ V^3
      Tct <- count (t, tc) #initial
    for t ∈ V^3
      prob[t][c] <- (Tct + 1) / Σ(Tct'+1) ; return V^3, prior, prob }
```

```
. test (C, V^3, prior, prob, x) {
  W <- get token {V^3, x}
  for c ∈ C
    score(c) <- log(prior(c))
    for t ∈ W
      score(c) += log(prob[t][c])
  return argmax_c score(c)
```

5241 hw5 Haiqi Li hl3115

Haiqi Li

13/4/2018

```r
H<-matrix(readBin("histograms.bin", "double", 640000), 40000, 16)
dim(H)
```

```
## [1] 40000    16
```

```r
H <- H+0.01 #avoid numerical problem
centroids_init <- function(K,H){
  # Initialization of centroid matrix T
  # args:
  # K: num of clusters
  # H: Histogram matrix
  #
  # returns:
  # T.matrix: A matrix of centroids.Row is centroid vectors
  choice <- sample(nrow(H),K,replace = F)
  T.matrix <-H[choice,]
  return(T.matrix)
}
```

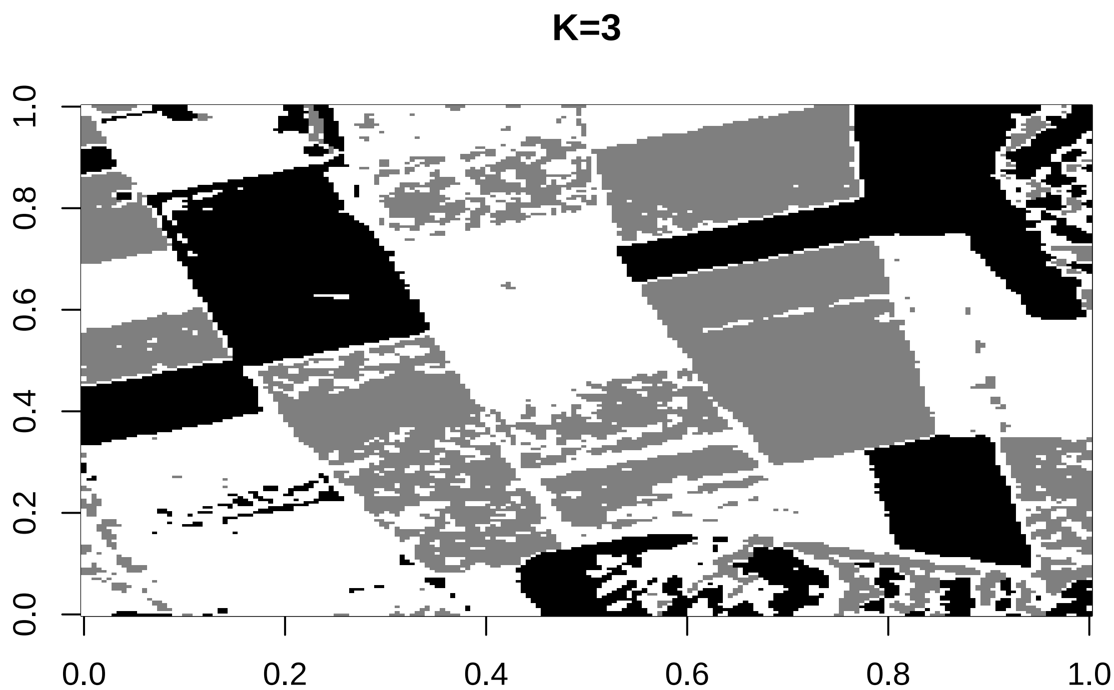
```r
E.step <- function(H,T.matrix,C){
  # E-step implementation
  # args:
  # H:n by d
  # T.matrix:k by d
  # C:k by 1 matrix,not vector
  # returns:
  # A:n by k
  phi <- exp(H %*% log(t(T.matrix)))
  A <- matrix(0,nrow = nrow(H),ncol = nrow(T.matrix))#init of A

  for (i in 1:nrow(H)) {
    dinominator <- (phi[i,] %*% C)
    for (k in 1:nrow(T.matrix)) {
      A[i,k] <- C[k,1]*phi[i,k]/dinominator
    }
  }

  return(A)
}
```

```r
M.step <- function(A,H){
  # implementation of M-step
  # args:
  #   A:n by k
  #   H:n by d
  # returns:
  # a list of (C,T.matrix)
  # C:k by 1 matrix
```

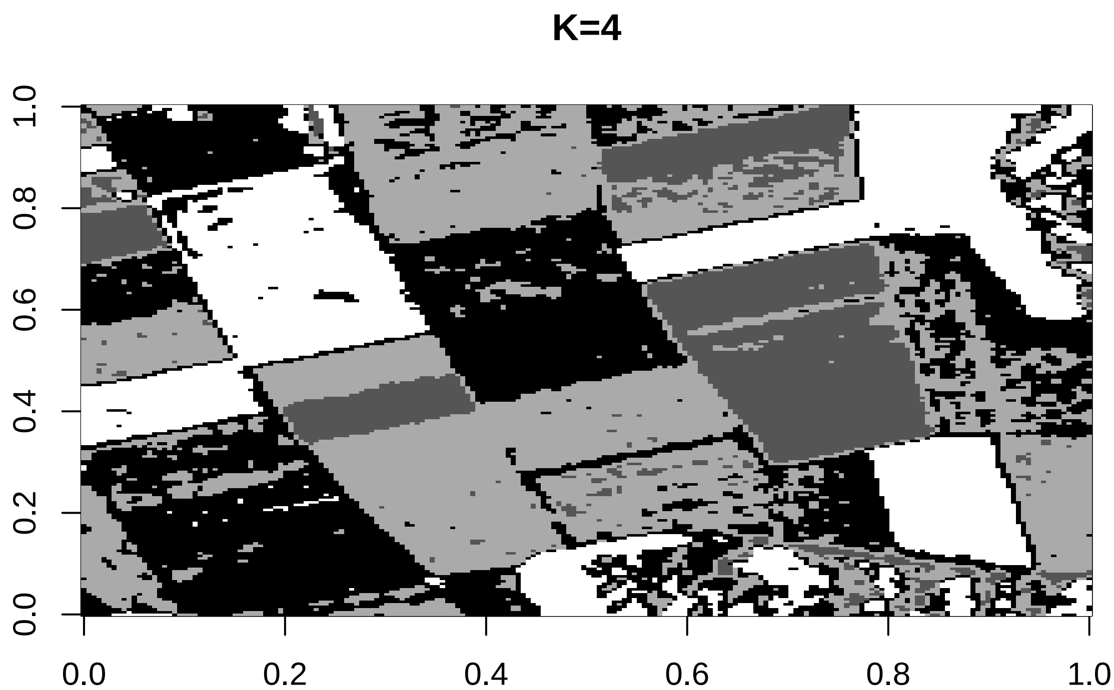
```r
  # T.matrix:k by d

  C <- matrix(colSums(A)/nrow(A),ncol=1)
  #C is k by 1 matrix
  b <- t(A) %*% H
  #b is k by d matrix,every row is b_k in hw
  row.normal <- function(row){
    sum.row <- as.numeric(sum(row))
    row <- row/sum.row
    return(row)
  }
  # a self-define funcyion to apply every row
  # with the dominate as sum of all rows

  T.matrix <- t(apply(b, 1, row.normal))
  out <- list(C=C,T.matrix=T.matrix)
  return(out)
}
```

```r
MultinomialEM <- function(H,K,tau){
  delta <- Inf
  T.matrix <- centroids_init(K,H)
  # The first step
  C <- matrix(1,nrow = K,ncol=1)

  A.prev <- E.step(H,T.matrix,C)


  while (delta>= tau) {
    temp <- M.step(A.prev,H)
    C <- temp$C
    T.matrix <- temp$T.matrix

    A <- E.step(H,T.matrix,C)
    delta <- norm(A-A.prev,"O")
    A.prev <- A
  }
  m <- apply(A.prev, 1, which.max)
  return(m)
}
```

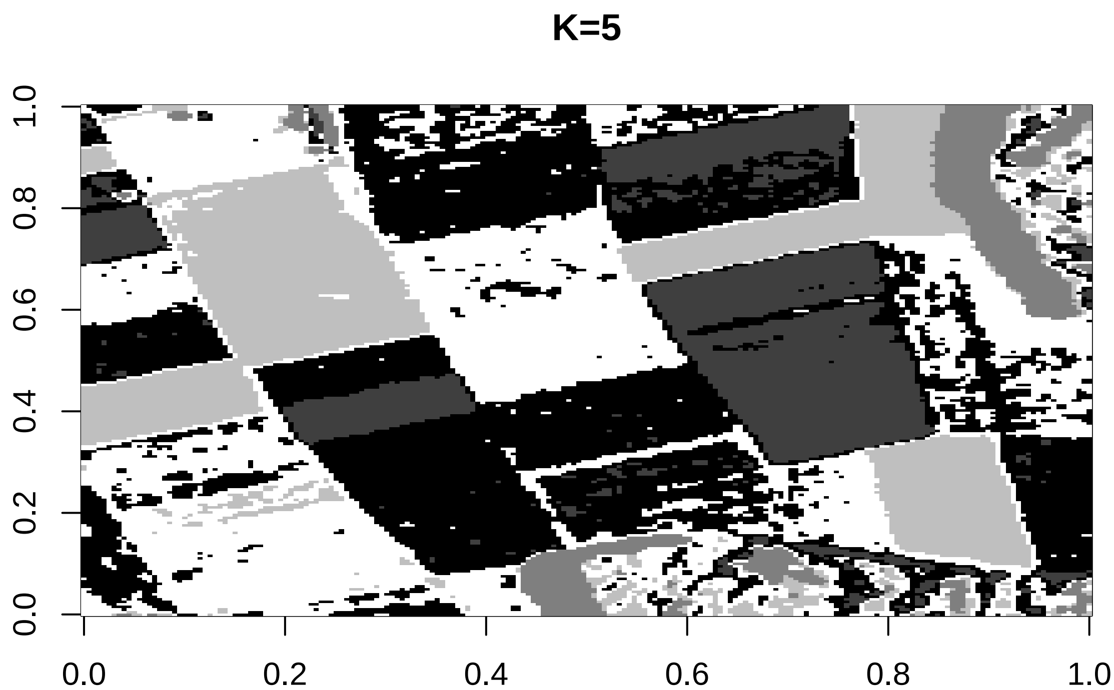
```r
set.seed(1)
m3 <- MultinomialEM(H,3,0.01)
pic3 <- matrix(m3,nrow = 200, ncol = 200, byrow = TRUE)
image(pic3, col = grey(seq(0, 1, length = 256)), main = "K=3")
```




```
m4 <- MultinomialEM(H,4,0.01)
pic4 <- matrix(m4,nrow = 200, ncol = 200, byrow = TRUE)
image(pic4, col = grey(seq(0, 1, length = 256)), main = "K=4")
```




```r
m5 <- MultinomialEM(H,5,0.01)
pic5 <- matrix(m5,nrow = 200, ncol = 200, byrow = TRUE)
image(pic5, col = grey(seq(0, 1, length = 256)), main = "K=5")
```


K=5
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0