10-701 Machine Learning, Fall 2012: Homework 3 Solutions

1 Clustering [35 points, Martin]

1.1 [25 points] K-means

In K-means clustering, we are given points $x_1, ..., x_n \in \mathbb{R}^d$ and an integer $K > 1$, and our goal is to minimize the within-cluster sum of squares (also known as the k-means objective)

$$J(C, L) = \sum_{i=1}^{n} \|x_i - C_{\ell_i}\|^2$$

where $C = (C_1, ..., C_K)$ are the cluster centers ($C_j \in \mathbb{R}^d$), and $L = (\ell_1, ..., \ell_n)$ are the cluster assignments ($\ell_i \in \{1, ..., K\}$).

Finding the exact minimum of this function is computationally difficult. The most common algorithm for finding an approximate solution is Lloyd's algorithm, which takes as input the set of points and some initial cluster centers C, and proceeds as follows:

i. Keeping C fixed, find cluster assignments L to minimize $J(C, L)$. This step only involves finding nearest neighbors. Ties can be broken using arbitrary (but consistent) rules.

ii. Keeping L fixed, find C to minimize $J(C, L)$. This is a simple step that only involves averaging points within a cluster.

iii. If any of the values in L changed from the previous iteration (or if this was the first iteration), repeat from step i.

iv. Return C and L.

The initial cluster centers C given as input to the algorithm are often picked randomly from $x_1, ..., x_n$. In practice, we often repeat multiple runs of Lloyd's algorithm with different initializations, and pick the best resulting clustering in terms of the k-means objective. You're about to see why.

(a) **[3 points]** Briefly explain why Lloyd's algorithm is always guaranteed to converge (i.e. stop) in a finite number of steps.

Answer: The cluster assignments L can take finitely many values (K^n, to be precise). The cluster centers C are uniquely determined by the assignments L, so after executing step ii the algorithm can be in finitely many possible states. Thus either the algorithm stops in finitely many steps, or at least one value of L is repeated more than once in non-consecutive iterations. However, the latter case is not possible, since after every iteration we have $J(C^{(t)}, L^{(t)}) \geq$

$J(C^{(t+1)}, L^{(t+1)})$, with equality only when $L^{(t)} = L^{(t+1)}$, which coincides with the termination condition. (Note that this statement depends on the assumption that the tie-breaking rule used in step i is consistent, otherwise infinite loops are possible.)

(b) **[5 points]** Implement Lloyd's algorithm. Run it until convergence 200 times, each time initializing using K cluster centers picked at random from the set $\{x_1, ..., x_n\}$, with $K = 5$ clusters, on the 500 two dimensional data points in `http://www.cs.cmu.edu/~epxing/Class/10701/HW/hw3-cluster.csv`. Plot in a single figure the original data (in gray), and all 200×5 cluster centers (in black) given by each run of Lloyd's algorithm. You can play around with the plotting options such as point sizes so that the cluster centers are clearly visible. Also compute the minimum, mean, and standard deviation of the within-cluster sums of squares for the clusterings given by each of the 200 runs.

Answer: Minimum: 222.37, mean: 249.66, standard deviation: 65.64. Plot in Figure 1. R code:

```
require(fields)

lloyd <- function(X,K) {
C <- X[sample.int(nrow(X), size = K, replace = FALSE),]
L <- rep(0,nrow(X))
L.old <- rep(-1,nrow(X))
while(any(L!=L.old)) {
L.old <- L
L <- apply(rdist(X,C),1,which.min)
C <- t(sapply(1:K,function(i) apply(subset(X,L==i),2,mean)))
}
return(list(cluster=L, centers=C))
}

within.sum.squares <- function(X,clustering) {
return(sum((X-clustering$centers[clustering$cluster,])^2))
}

X <- as.matrix(read.csv("hw3-cluster.csv",header=F))
dimnames(X)[[2]]<-NULL

nstart <- 200
K <- 5
centers.all <- c()
within.ss <- rep(0,nstart)
for(i in 1:nstart) {
cs <- lloyd(X,5)
centers.all <- rbind(centers.all,cs$centers)
within.ss[i] <- within.sum.squares(X,cs)
}

cat(paste("Minimum: ",round(min(within.ss),2),", mean: ",round(mean(within.ss),2),",
 standard deviation: ",round(sd(within.ss),2),"\n",sep=""))
```

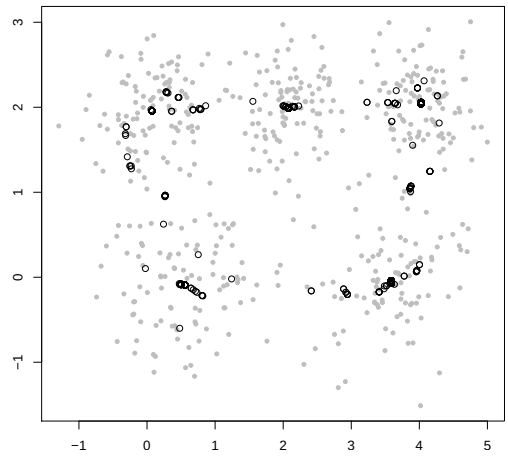


Figure 1

```
plot(X,col=8,pch=20,xlab="",ylab="")
points(centers.all)
```

(c) **[4 points]** Kmeans++ is an initialization algorithm for K-means proposed by David Arthur and Sergei Vassilvitskii in 2007:

i. Pick the first cluster center C_1 uniformly at random from the data $x_1, ..., x_n$. In other words, we first pick an index i uniformly at random from $\{1, ..., n\}$, then set $C_1 = x_i$.

ii. For $j = 2, ..., K$:

- For each data point, compute its distance D_i to the nearest cluster center picked in a previous iteration:
$$D_i = \min_{j'=1,...,j-1} \|x_i - C_{j'}\|.$$
- Pick the cluster center C_j at random from $x_1, ..., x_n$ with probabilities proportional to $D_1^2, ..., D_n^2$. Precisely, we pick an index i at random from $\{1, ..., n\}$ with probabilities equal to $D_1^2/(\sum_{i'=1}^n D_{i'}^2), ..., D_n^2/(\sum_{i'=1}^n D_{i'}^2)$, and set $C_j = x_i$.

iii. Return C as the initial cluster assignments for Lloyd's algorithm.

Replicate the figure and calculations in part (b) using Kmeans++ as the initialization algorithm, instead of picking C uniformly at random.

Answer: Minimum: 222.37, mean: 248.33, standard deviation: 64.96. Plot in Figure 2. R code:

```
lloyd.kmeanspp <- function(X,K) {
C <- rbind(X[sample.int(nrow(X), size = 1),])
for(j in 2:K) {
C <- rbind(C,X[sample.int(nrow(X),size=1,prob=apply(rdist(X,C),1,min)^2),])
}
L <- rep(0,nrow(X))
L.old <- rep(-1,nrow(X))
```

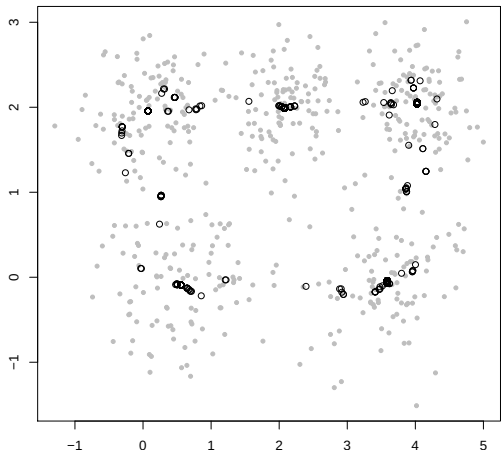


Figure 2

```
while(any(L!=L.old)) {
L.old <- L
L <- apply(rdist(X,C),1,which.min)
C <- t(sapply(1:K,function(i) apply(subset(X,L==i),2,mean)))
}
return(list(cluster=L, centers=C))
}

centers.all <- c()
within.ss <- rep(0,nstart)
for(i in 1:nstart) {
cs <- lloyd(X,5)
centers.all <- rbind(centers.all,cs$centers)
within.ss[i] <- within.sum.squares(X,cs)
}

cat(paste("Minimum: ",round(min(within.ss),2),", mean: ",round(mean(within.ss),2),",
 standard deviation: ",round(sd(within.ss),2),"\n",sep=""))

plot(X,col=8,pch=20,xlab="",ylab="")
points(centers.all)
```

Hopefully your results make it clear how sensitive Lloyd's algorithm is to initializations, even in such a simple, two dimensional data set!

Picking the number of clusters K is a difficult problem. Now we will see one of the most common heuristics for choosing K in action.

(d) **[3 points]** Explain how the exact minimum of the k-means objective behaves on any data set as we increase K from 1 to n.

Answer: The exact minimum decreases (or stays the same) as K increases, because the set of possible clusterings for K is a subset of the possible clusterings for $K+1$. With $K=n$, the objective of the optimal solution is 0 (every point is in its own cluster, and has 0 distance

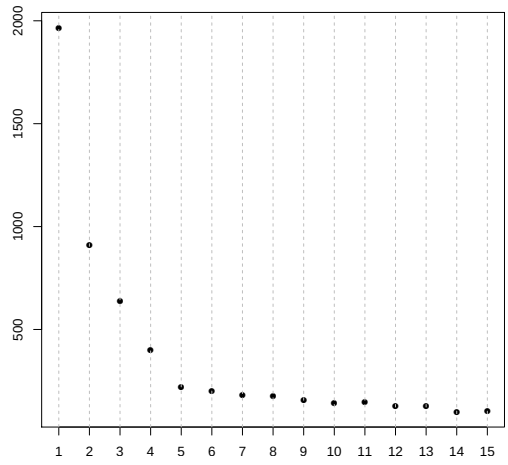


Figure 3

to the cluster center).

A common way to pick K is as follows. For each value of K in some range (e.g. $K = 1, ..., n$, or some subset), we find an approximate minimum of the k-means objective using our favorite algorithm (e.g. multiple runs of randomly initialized Lloyd's algorithm). Then we plot the resulting values of the k-means objective against the values of K. Often, if our data set is such that there exists a natural value for K, we see a "knee" in this plot, i.e. a value for K where the rate at which the within-cluster sum of squares is decreasing sharply reduces. This suggests we should use the value for K where this knee occurs. In the toy example in Figure 5, this value would be $K = 6$.

(e) **[3 points]** Produce a plot similar to the one in Figure 5 for $K = 1, ..., 15$ using the data set in (b), and show where the "knee" is. For each value of K, run k-means with at least 200 initializations and pick the best resulting clustering (in terms of the objective) to ensure you get close to the global minimum.

Answer: Plot in Figure 3. The knee is at $K = 5$.

(f) **[3 points]** Repeat part (e) with the data set in `http://www.cs.cmu.edu/~epxing/Class/10701/HW/hw3-cluster2.csv`. Find 2 knees in the resulting plot (you may need to plot the square root of the within-cluster sum of squares instead, in order to make the second knee obvious). Explain why we get 2 knees for this data set (consider plotting the data to see what's going on).

Answer: Plot in Figure 4 (square root of objective plotted). The knees are at $K = 3$ and $K = 9$. These are two values because the data are composed of 3 natural clusters, each of which can further be divided into 3 smaller clusters.

We conclude our exploration of k-means clustering with the critical importance of properly scaling the dimensions of your data.

(g) **[2 points]** Load the data in `http://www.cs.cmu.edu/~epxing/Class/10701/HW/hw3-cluster3.csv`. Perform k-means clustering on this data with $K = 2$ with 500 initializations. Plot the original data (in gray), and overplot the 2 cluster centers (in black).

Answer: See Figure 6.

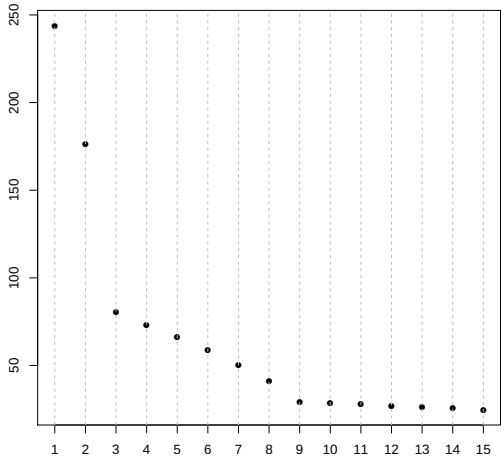


Figure 4

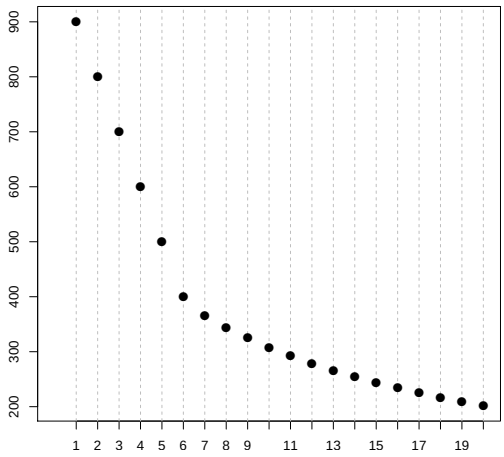


Figure 5: Picking the number of clusters (a toy example).

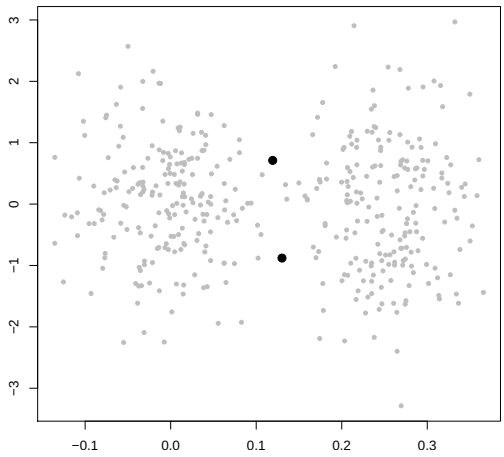


Figure 6

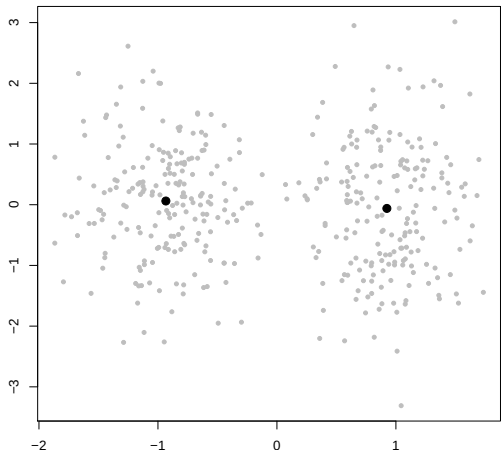


Figure 7

(h) **[2 points]** Normalize the features in this data set, i.e. first center the data to be mean 0 in every dimension, then rescale each dimension to have unit variance. Repeat part (g) with this modified data.

Answer: See Figure 7.

As you can see, the results are radically different. You should not take this to mean that data should *always* be normalized. In some problems, the relative values of the dimensions are meaningful and should be preserved (e.g. the coordinates of earthquake epicenters in a region). But in others, the dimensions are on entirely different scales (e.g. age in years v.s. income in thousands of dollars). Proper pre-processing of data for clustering is often part of the art of machine learning.

1.2 Hierarchical clustering [10 points]

Agglomerative hierarchical clustering is a family of hierarchical clustering algorithms that, equipped with a notion of distance between clusters, form a binary tree with leaves for each original data point as follows:

i. Initialize by placing each data point in its own cluster (i.e. singleton trees).

ii. Find the two closest clusters, join them in a single cluster (by creating a new node and making it the parent of the roots of those two clusters).

iii. If there are more than one clusters (trees) left, repeat from step i.

iv. Return the final tree.

Some of the most common metrics of distance between two clusters $\{x_1, ..., x_m\}$ and $\{y_1, ..., y_p\}$ are:

- *Single linkage*: Distance between clusters is the *minimum* distance between any pair of points from the two clusters, i.e.

$$\min_{\substack{i=1,...,m \\ j=1,...,p}} \|x_i - y_j\|;$$

- *Complete linkage*: Distance between clusters is the *maximum* distance between any pair of points from the two clusters, i.e.

$$\max_{\substack{i=1,\ldots,m \\ j=1,\ldots,p}} \|x_i - y_j\|;$$

- *Average linkage*: Distance between clusters is the *average* distance between all pair of points from the two clusters, i.e.

$$\frac{1}{m \cdot p} \sum_{i=1}^{m} \sum_{j=1}^{p} \|x_i - y_j\|.$$

Also, given a clustering tree, we can define a partitioning of the data into K clusters by "cutting" the tree some number of levels below the root. For example, if $K = 2$ we could define two clusters based on the left and right subtrees of the root of the clustering tree. If $K = 4$, we could use the subtrees of the children of the root, etc. (Note that if K is not a power of 2 we would need to come up with some way of deciding which subtree gets preference.)

(a) **[3 points]** Using this procedure for turning a hierarchical clustering into a partition, which of the three cluster similarity metrics described above would most likely result in clusters most similar to those given by k-means? (Assume K is a power of 2).

Answer: Average linkage.

(b) **[4 points]** Consider the data in Figure 8a. What would be the result if we extracted $K = 2$ clusters from the tree given by hierarchical clustering on this data set using single linkage? (Describe your answer in terms of the labels $1 - 4$ given to the four "clumps" in the data.) Do the same for complete and average linkage.

Answer: Average and complete linkage would assign "clumps" 1 and 3 to the first cluster, and 2 and 4 to the second. Single linkage would assign 1 and 2 to one cluster, 3 and 4 to the other.

(c) **[3 points]** Which of those three distance metrics (if any) would successfully separate the two "moons" in Figure 8b? What about Figure 8c? Briefly explain your answer.

Answer: Single linkage would successfully separate the two moons in Figure 8b, average and complete linkage would not. None of the methods would work in Figure 8c.

2 Bayesian Network [25 points, Avi]

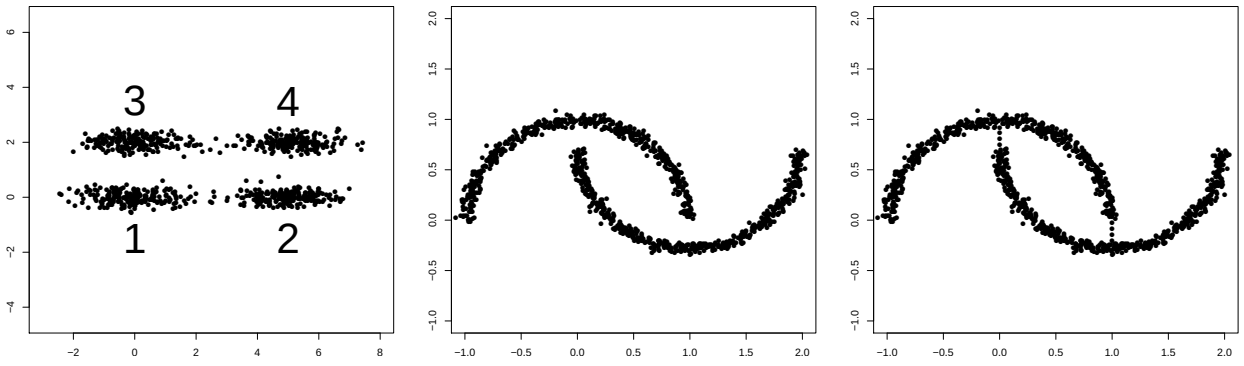
This problem will concern the Bayesian network in Figure 9.

2.1 [3 points] Joint Probability

Write down the factorization of the joint probability distribution over A, B, C, D, E, F which corresponds to this graph.

Answer:-

$$P(A, B, C, D, E, F, G) = P(A)P(B|A)P(C)P(D|B)P(E|B, C)P(F|D, E) \tag{1}$$



(a) (b) (c)

Figure 8

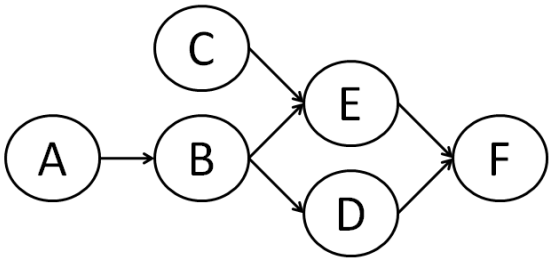


Figure 9

2.2 [5 points] conditional independence

For every pair of nodes in the graph say whether each of them are independent of each other or not. Also test conditional independence for each pair except F when F is observed

Answer:- (A, C), (B, C) and (C, D) are independent when nothing is observed. Nothing is independent when F is observed

2.3 [12 points] Inference

For this section we suppose that all the variables are binary, taking on the values $0, 1$. The conditional probability distributions on the graph have the following form:

- Nodes with a single parent take the value of their parent with probability $\frac{3}{4}$ otherwise they take the other value.
- Nodes with two parents take the value of the first parent with probability $\frac{1}{2}$ otherwise they take the value of the second parent.
- $P(A = 1) = p$, $P(C = 1) = q$.

All the assumptions made in this section holds for all the three questions below

1. **[2 points] CPT Tricks I**. If some node X has a single parent Y, and $P(Y = 1) = a$, what is a simple expression for $P(X = 1)$? Please assume that there is no child of X to worry about.

Answer:-

$$\begin{aligned} P(X=1) &= P(X=1|Y=0)P(Y=0) + P(X=1|Y=1)P(Y=1) \\ &= \frac{1}{4}(1-a) + \frac{3}{4}a \\ &= \frac{1}{4} + \frac{1}{2}a \end{aligned}$$

2. **[3 points] CPT Tricks II**. If some node X has a two independent parents Y, Z, and $P(Y = 1) = a,\ P(Z = 1) = b$, what is a simple expression for $P(X = 1)$? Please assume that there is no child of X to worry about.

   Answer:-

$$\begin{aligned} P(X=1) &= P(X=1|Y=0,Z=0)P(Y=0)P(Z=0) + P(X=1|Y=1,Z=0)P(Y=1)P(Z=0) \\ &\quad P(X=1|Y=0,Z=1)P(Y=0)P(Z=1) + P(X=1|Y=1,Z=1)P(Y=1)P(Z=1) \\ &= 0 + \frac{1}{2}a(1-b) + \frac{1}{2}(1-a)b + ab \\ &= \frac{1}{2}a + \frac{1}{2}b \end{aligned}$$

3. **[7 points] Forwards Inference**. What is $P(F = 1)$ in the above graph?

   Answer:-

$$\begin{aligned} P(B=1) &= \frac{1}{4} + \frac{1}{2}p \\ P(D=1) &= \frac{1}{4} + \frac{1}{2}P(B=1) \\ &= \frac{3}{8} + \frac{1}{4}p \\ P(E=1) &= \frac{1}{2}P(B=1) + \frac{1}{2}P(C=1) \\ &= \frac{1}{8} + \frac{1}{4}p + \frac{1}{2}q \\ P(F=1) &= \frac{1}{2}P(D=1) + \frac{1}{2}P(E=1) \\ &= \frac{3}{16} + \frac{1}{8}p + \frac{1}{16} + \frac{1}{8}p + \frac{1}{4}q \\ &= \frac{1}{4} + \frac{1}{4}p + \frac{1}{4}q \end{aligned}$$

2.4 [5 points] Conditional Inference

If $B = b, F = f$ are observed, what is the conditional probability that $E = 1$? For this question please leave your answer in terms of probability distributions e.g., $P(B = b|A = a)$ etc., but only those which could be computed directly from the local probabilities in the definition of the Bayes net.

Answer:-

$$P(E=1|B=b,F=f)=\sum_{c=0}^{1}\frac{\sum_d P(F=f|D=d,E=e)P(D=d|B=b)P(E=e|B=b,C=c)}{\sum_e\sum_d P(F=f|D=d,E=e)P(D=d|B=b)P(E=e|B=b,C=c)}P(C=c)$$

3 Expectation Maximization [20pt, Derry]

1. **[15 points]** In this question, you are going to derive the Expectation and Maximization equations of the EM algorithm for optimizing the latent variables involved in generating a text document.

   Consider each word as a random variable w that can take values $1, ..., V$ from the vocabulary of words. Treat each w as a vector of $|V|$ components such that $w(i) = 1$ if the w takes the value of the i^{th} word in the vocabulary. Hence, $\sum_i^V w(i) = 1$. The words are generated from a mixture of M discrete topics:

   $$p(w) = \sum_{m=1}^{M} \pi_m p(w|\mu_m)$$

   and

   $$p(w|\mu_m) = \prod_{i=1}^{V} \mu_m(i)^{w(i)}$$

   where π_m denotes the prior for the latent topic variable $t = m$ and $\mu_m(i) = p(w(i) = 1|t = m)$, thus $\sum_{i=1}^{V} \mu_m(i) = 1$.

   Given a document containing words w_j, $j = 1, ..., N$, where N is the length of the document, **derive** the expectation and maximization step equations for the EM algorithm to optimize π_m and $\mu_m(i)$.

   Note: Show all the steps in your derivation.

   Hints:

   - In the expectation step **[5 points]** , for each word w_j, compute $F_j(t_j) = p(t_j|w_j; \theta)$, the probability that w_j belongs to each of the M topic.

     Answer:

$$
\begin{aligned}
F_j(t_j = m) &= p(t_j = m|w_j; \theta) \\
&= \frac{p(w_j|t_j = m; \theta)p(t_j = m|\theta)}{p(w_j|\theta)} \\
&= \frac{\pi_m p(w_j|\mu_m)}{\sum_{m'=1}^{M} \pi_{m'} p(w_j|\mu_{m'})} \\
&= \frac{\pi_m \prod_{l=1}^{V} \mu_m(l)^{w_j(l)}}{\sum_{m'=1}^{M} \pi_{m'} \prod_{l=1}^{V} \mu_{m'}(l)^{w_j(l)}}
\end{aligned}
$$

- In the maximization step **[10 points]**, compute θ which is the set of parameters of this mixture model that maximizes the log likelihood of the data

$$l(w; \theta) = log \prod_{j=1}^{N} p(w_j; \theta)$$

Summing over the latent topic variable:

$$l(w; \theta) = \sum_{j=1}^{N} log \sum_{t_j} p(w_j, t_j; \theta)$$

$$l(w; \theta) = \sum_{j=1}^{N} log \sum_{t_j} F_j(t_j) \frac{p(w_j, t_j; \theta)}{F_j(t_j)}$$

Using Jensen's inequality:

$$l(w; \theta) \geq \sum_{j=1}^{N} \sum_{t_j} F_j(t_j)\ log\ \frac{p(w_j, t_j; \theta)}{F_j(t_j)} = \sum_{j=1}^{N} \sum_{t_j} F_j(t_j)\ log\ p(w_j; \theta) = \sum_{j=1}^{N} log\ p(w_j; \theta) = l(w; \theta)$$

Hence compute θ as:

$$\theta := argmax_\theta \sum_{j=1}^{N} \sum_{t_j} F_j(t_j)\ log\ \frac{p(w_j, t_j; \theta)}{F_j(t_j)}$$

Answer:

$$
\begin{aligned}
\theta &:= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} F_j(t_j = m)\ log\ \frac{p(w_j, t_j = m; \theta)}{F_j(t_j = m)} \\
&= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} F_j(t_j = m)\ log\ p(w_j, t_j = m; \theta) \\
&= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} F_j(t_j = m)\ log\ p(w_j | t_j = m; \theta)\ p(t_j = m; \theta) \\
&= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} F_j(t_j = m)\ log\ \pi_m \prod_{l=1}^{V} \mu_m(l)^{w_j(l)} \\
&= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} \left(F_j(t_j = m)\ log\ \pi_m + F_j(t_j = m) \sum_{l=1}^{V} log\ \mu_m(l)^{w_j(l)} \right) \\
&= argmax_\theta \sum_{j=1}^{N}\sum_{m=1}^{M} \left(F_j(t_j = m)\ log\ \pi_m + F_j(t_j = m) \sum_{l=1}^{V} w_j(l)\ log\ \mu_m(l) \right)
\end{aligned}
$$

To optimize $\mu_m(l)$: first eliminate terms that are constant with respect to μ_m:

$$
\sum_{j=1}^{N} F_j(t_j = m) \sum_{l=1}^{V} w_j(l)\ log\ \mu_m(l)
$$

Use a Lagrangian to constrain μ_m to be a probability distribution:

$$
\mathcal{L}(\mu_m(l)) = \sum_{j=1}^{N} F_j(t_j = m) \sum_{l=1}^{V} w_j(l)\ log\ \mu_m(l) + \beta \left(\sum_{l=1}^{V} \mu_m(l) - 1 \right)
$$

Solving for $\mu_m(l)$:

$$
\begin{aligned}
\frac{\partial}{\partial_{\mu_m(l)}} \mathcal{L}(\mu_m(l)) = \sum_{j=1}^{N} F_j(t_j = m)\ \frac{w_j(l)}{\mu_m(l)} + \beta &= 0 \\
\frac{1}{\mu_m(l)} \sum_{j=1}^{N} F_j(t_j = m)\ w_j(l) + \beta &= 0 \\
\frac{1}{\mu_m(l)} &= \frac{-\beta}{\sum_{j=1}^{N} F_j(t_j = m)\ w_j(l)} \\
\mu_m(l) &= \frac{\sum_{j=1}^{N} F_j(t_j = m)\ w_j(l)}{-\beta}
\end{aligned}
$$

Knowing that $\sum_{l=1}^{V} \mu_m(l) = 1$ we have:

$$\begin{aligned}
\sum_{l=1}^{V} \frac{\sum_{j=1}^{N} F_j(t_j = m)\, w_j(l)}{-\beta} - 1 &= 0 \\
\frac{1}{-\beta} \sum_{l=1}^{V} \sum_{j=1}^{N} F_j(t_j = m)\, w_j(l) &= 1 \\
\sum_{l=1}^{V} \sum_{j=1}^{N} F_j(t_j = m)\, w_j(l) &= -\beta
\end{aligned}$$

Hence, substituting for $-\beta$:

$$\mu_m(l) = \frac{\sum_{j=1}^{N} F_j(t_j = m)\, w_j(l)}{\sum_{l=1}^{V} \sum_{j=1}^{N} F_j(t_j = m)\, w_j(l)}$$

Since $\sum_{l=1}^{V} w_j(l) = 1$:

$$\mu_m(l) = \frac{\sum_{j=1}^{N} F_j(t_j = m)\, w_j(l)}{\sum_{j=1}^{N} F_j(t_j = m)}$$

Intuitively this can be interpreted as the portion that had $w(l) = 1$ among the mass that was deemed to belong to cluster m.

Similarly, to optimize π_m, begin by removing terms that are constant with respect to π_m:

$$\sum_{j=1}^{N} F_j(t_j = m)\, log\, \pi_m$$

Using the Lagrangian with the constraint that $\sum_{m=1}^{M} \pi_m = 1$

$$\mathcal{L}(\pi_m) = \sum_{j=1}^{N} F_j(t_j = m)\, log\, \pi_m + \beta \left(\sum_{m=1}^{M} \pi_m - 1 \right)$$

Solving for π_m:

$$\begin{aligned}
\frac{\partial}{\partial_{\pi_m}}\mathcal{L}(\pi_m) = \sum_{j=1}^{N}\frac{F_j(t_j=m)}{\pi_m} + \beta &= 0 \\
\frac{1}{\pi_m}\sum_{j=1}^{N}F_j(t_j=m) &= -\beta \\
\sum_{j=1}^{N}F_j(t_j=m) &= -\beta\,\pi_m \\
\pi_m &= \frac{\sum_{j=1}^{N}F_j(t_j=m)}{-\beta}
\end{aligned}$$

Since $\sum_{m=1}^{M}\pi_m = 1$:

$$\begin{aligned}
\sum_{m=1}^{M}\frac{\sum_{j=1}^{N}F_j(t_j=m)}{-\beta} &= 1 \\
\frac{1}{-\beta}\sum_{m=1}^{M}\sum_{j=1}^{N}F_j(t_j=m) &= 1 \\
-\beta &= \sum_{m=1}^{M}\sum_{j=1}^{N}F_j(t_j=m)
\end{aligned}$$

Substituting for $-\beta$ we get:

$$\pi_m = \frac{\sum_{j=1}^{N}F_j(t_j=m)}{\sum_{m=1}^{M}\sum_{j=1}^{N}F_j(t_j=m)} = \frac{\sum_{j=1}^{N}F_j(t_j=m)}{N}$$

Intuitively this can be interpreted as the portion that belongs to cluster m among the total of N examples.

4 Hidden Markov Model [20 points, Zeyu]

1. Backward probability
2. Parameters: emission probability $a_{i,j}$, transition probability $b_{i,k}$ and starting state probability π_i for all $i, j = 1, 2, ..., K$ and $k = 1, 2, ..., M$
3. The complete likelihood function

$$P(X, Y|\Theta) = p(y_1)\prod_{t=2}^{T}p(y_t|y_{t-1})p(x_t|y_t) \tag{2}$$

$$
\begin{aligned}
\beta_t^k &= P(x_{t+1},...,x_T \mid y_t^k = 1) \\
&= \sum_{y_{t+1}} P(, y_{t+1}, x_{t+1},...,x_T \mid y_t^k = 1) \\
&= \sum_i P(y_{t+1}^i = 1 \mid y_t^k = 1) p(x_{t+1} \mid y_{t+1}^i = 1, y_t^k = 1) P(x_{t+2},...,x_T \mid x_{t+1}, y_{t+1}^i = 1, y_t^k = 1) \\
&= \sum_i P(y_{t+1}^i = 1 \mid y_t^k = 1) p(x_{t+1} \mid y_{t+1}^i = 1) P(x_{t+2},...,x_T \mid y_{t+1}^i = 1) \\
&= \sum_i a_{k,i} p(x_{t+1} \mid y_{t+1}^i = 1) \beta_{t+1}^i
\end{aligned}
$$

Write it in the suggested form

$$P(X, Y|\Theta) = (\prod_{i=1}^{K} \pi^{y_1^i}) \times (\prod_{t=2}^{T} \prod_{i=1}^{K} \prod_{j=1}^{K} a_{i,j}^{y_t^j y_{t-1}^i}) \times (\prod_{t=1}^{T} \prod_{i=1}^{K} \prod_{k=1}^{M} b_{i,k}^{x_t^k y_t^i}) \tag{3}$$

4. Take the log

$$log(P(X, Y|\Theta) = (\sum_{i=1}^{K} y_1^i \log \pi_i) + \sum_{t=2}^{T} \sum_{i=1}^{K} \sum_{j=1}^{K} y_t^j y_{t-1}^i \log a_{ij} + \sum_{t=1}^{T} \sum_{i=1}^{K} y_t^i \log b_{i,x_t} \tag{4}$$

And add the expectation

$$< log(P(X, Y|\Theta) >= (\sum_{i=1}^{K} < y_1^i > \log \pi_i) + \sum_{t=2}^{T} \sum_{i=1}^{K} \sum_{j=1}^{K} < y_t^j y_{t-1}^i > \log a_{ij} + \sum_{t=1}^{T} \sum_{i=1}^{K} < y_t^i > \log b_{i,x_t} \tag{5}$$

Note that $Q(\Theta, \Theta^{old}) =< log(P(X, Y|\Theta) >= \int_Y p(Z|X, \Theta) log P(X, Y|\Theta^{old})$

5. The so-called adjusted HMM in this question is actually a simplified IOHMM

   (a) Bayesian network of IOHMM is shown in Figure 10(a)

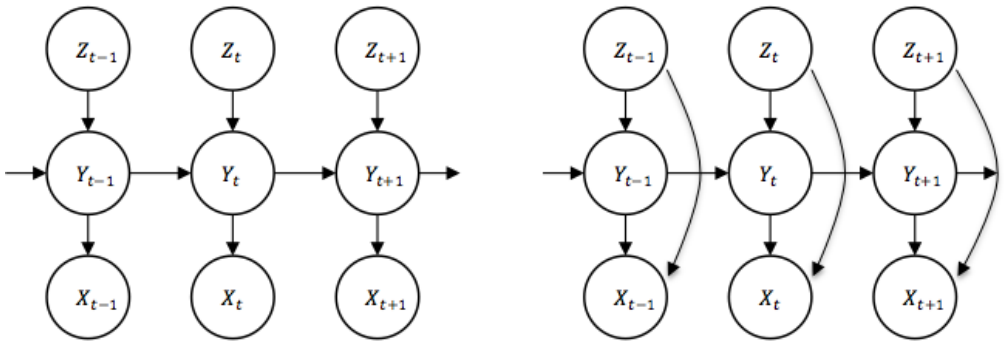


Figure 10: Bayesian network of an Input-output hidden markov model

   (b) Complete probability (likelihood) is

$$P(X, Y, Z|\Theta) = p(y_1|z_1)p(z_1) \prod_{t=2}^{T} p(z_t)p(y_t|y_{t-1}, z_t)p(x_t|y_t) \tag{6}$$

Note: many of you forgot $p(z_t)$. If without this term, an IOHMM will go back to an HMM (why?).

(c) how many states? mn (just think of automata multiplication). Answering n^2m transitions or coefficients is also acceptable.