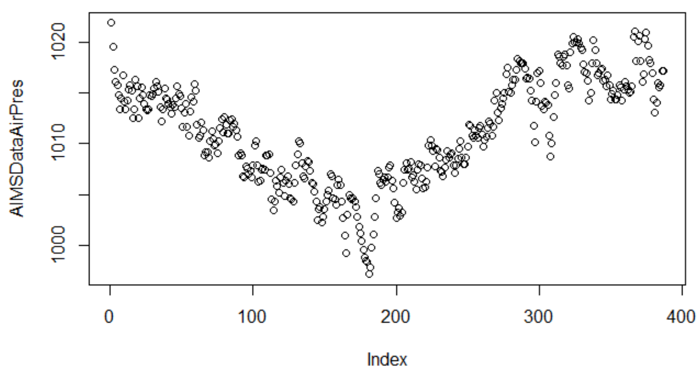
SIT743 Multivariate and Categorical Data Analysis
Assignment-2 (T2-2018)

Outline of ANSWERS

Q1)

1.1)

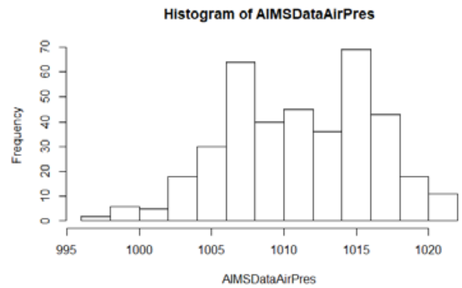
plot(AIMSDataAirPres)




```
> summary(AIMSDataAirPres)
   X1022.0056
 Min.   : 997.1
 1st Qu.:1007.2
 Median :1011.4
 Mean   :1011.1
 3rd Qu.:1015.3
 Max.   :1021.9

> sd(AIMSDataAirPres)
[1] 5.146871
```

1.2)



There are two modes in the distribution.

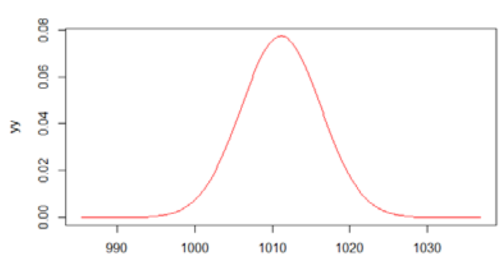
The distribution is slightly skewed to the left.

1.3)

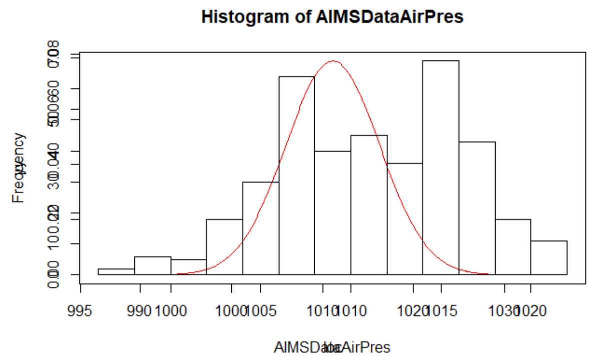
```
> library(MASS)
> fit1<-fitdistr(AIMSDataAirPres,"normal")
>  fit1$estimate
       mean           sd 
1011.139038    5.140217 
```

```
est<-fit1$estimate
est
m <- getElement(est,"mean")
s <- getElement(est,"sd")
loc<-c(seq(m-5*s,m+5*s,0.1))
loc
yy<-dnorm(loc,getElement(est,"mean"), getElement(est,"sd"))
yy
plot(loc, yy, col="red", 'l')

par(new=TRUE)
hist(AIMSDataAirPres)
```



OR



1.4)

#Gaussian mixture - 2 Gaussian

mixmdl = normalmixEM(AIMSDataAirPres,k=2)

mixmdl

summary(mixmdl)

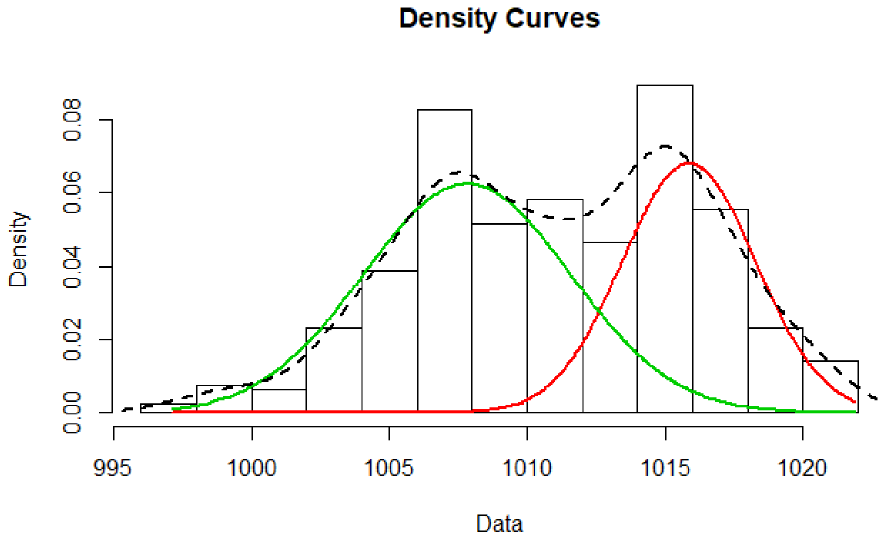
```
> summary(mixmdl)
summary of normalmixEM object:
            comp 1      comp 2
lambda    0.413058    0.586942
mu     1015.864975 1007.813192
sigma     2.425176    3.754725
loglik at estimate:  -1165.06
```

1.5)

par(mfrow=c(1,1))

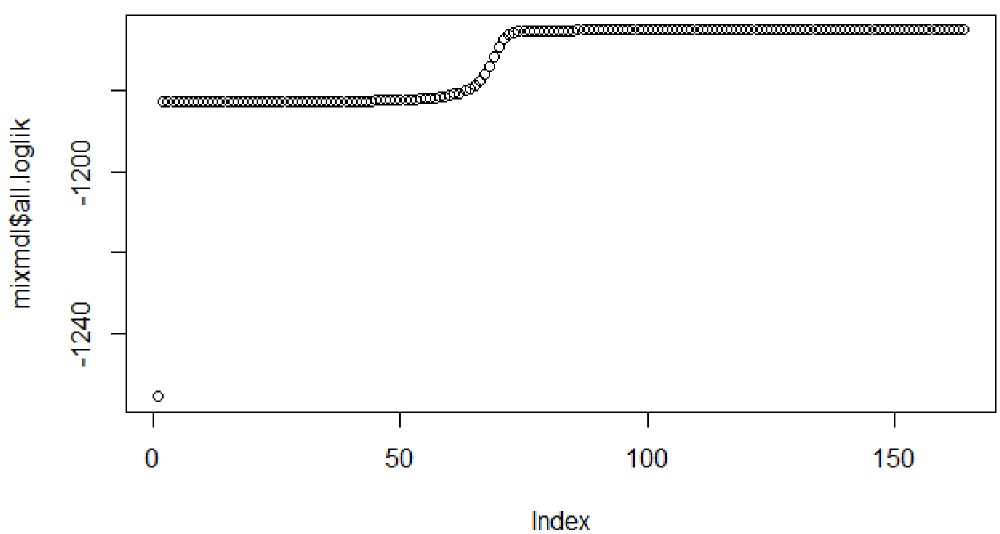
plot(mixmdl,which=2)

lines(density(AIMSDataAirPres), lty=2, lwd=2)



1.6)

plot(mixmdl$all.loglik)



Logliklihood values have stabilized after around 2th iteration and then after 72nd iteration.

1.7)

Gaussian mixture model with 2 Gaussians describes the distribution much better than using a single Gaussian distribution for this data since it has two modes.

1.8)

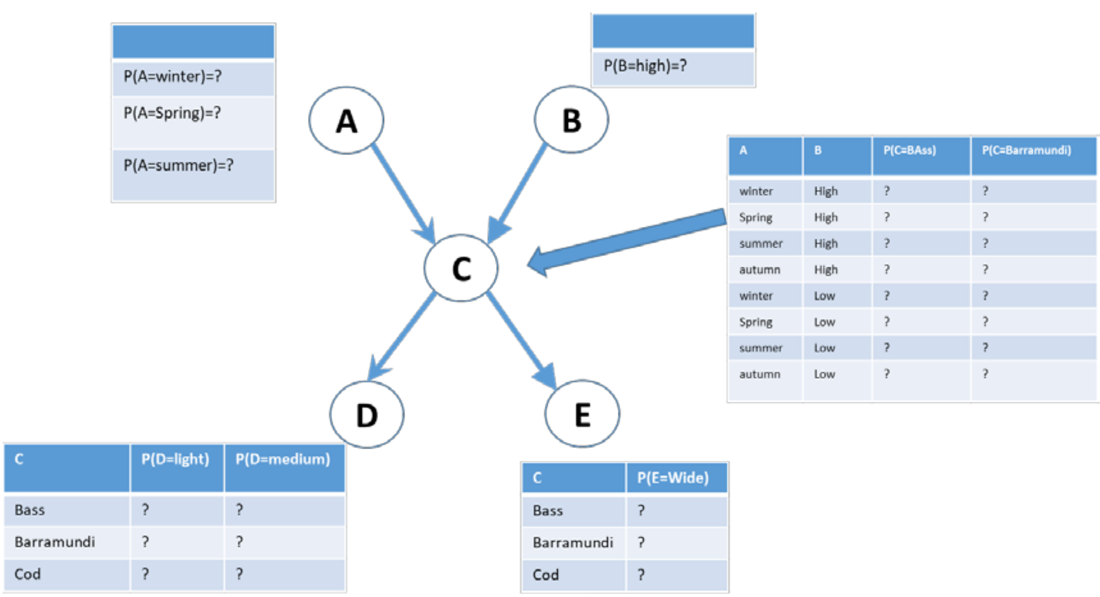
Presense of Singularities.
This might lead to occurrence of severe Overfitting. To overcome, use heuristics to detect this (by observing the mean and the standard deviation values during iterations) and reset the mean to a randomly chosen value while resetting its covariance to some large value, and then continue with the optimization.

Q2)

2.1)

$$p(A, B, C, D, E) = p(A)p(B)p(C|A, B)p(D|C)p(E|C)$$

2.2)



P(A=winter)=?
P(A=Spring)=?
P(A=summer)=?

P(B=high)=?

A	B	P(C=BAss)	P(C=Barramundi)
winter	High	?	?
Spring	High	?	?
summer	High	?	?
autumn	High	?	?
winter	Low	?	?
Spring	Low	?	?
summer	Low	?	?
autumn	Low	?	?

C	P(D=light)	P(D=medium)
Bass	?	?
Barramundi	?	?
Cod	?	?

C	P(E=Wide)
Bass	?
Barramundi	?
Cod	?

Number of parameters = $\underbrace{3}_{\boxed{A}} + \underbrace{1}_{\boxed{B}} + \underbrace{(4 * 2) * (3 - 1)}_{\boxed{C}} + \underbrace{3 * (3 - 1)}_{\boxed{D}} + \underbrace{3 * (2 - 1)}_{\boxed{E}} = 29$

2.3)

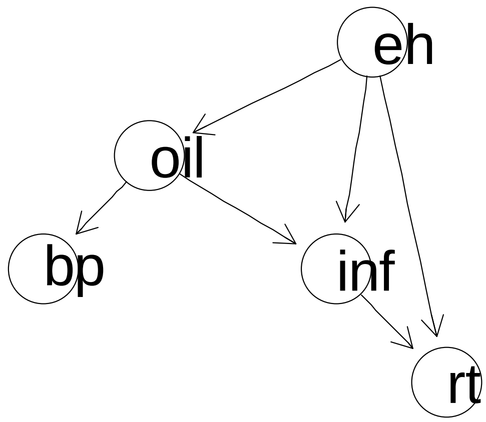
Number of parameters = $4 * 2 * 3 * 3 * 2 - 1 = 143$

The number of parameters required are far higher if no independencies among the variables is assumed

2.4)

$$P(A = \text{summer} \mid D = \text{dark}, E = \text{wide}) = \frac{\sum_{B,C} p(A=summer,B,C,D=dark,E=wide)}{\sum_{A,B,C} p(A,B,C,D=dark,E=wide)}$$

Q3)
3.1)



3.2)

```
library("gRain")

#source("https://bioconductor.org/biocLite.R")

#library("Rgraphviz")

source("https://bioconductor.org/biocLite.R")

#biocLite("RBGL")

library(RBGL)

library(gRbase)

library(gRain)


lh <- c("low","high")

lhn <- c("low","normal","high")

eh.o <- cptable(~eh, values=c(0.20,0.80),levels=lh)

bp.oil <- cptable(~bp|oil, values=c(0.80,0.15,0.05,0.1,0.4,0.5),levels=lhn)
```

oil.eh <- cptable(~oil|eh, values=c(0.90,0.1,0.05,0.95),levels=lh)

rt.inf.eh <-
cptable(~rt|inf:eh,values=c(0.6,0.30,0.10,0.2,0.2,0.6,0.1,0.2,0.7,0.05,0.1,0.85),levels=lhn)

inf.oil.eh <-
cptable(~inf|eh:oil,values=c(0.90,0.10,0.1,0.9,0.2,0.8,0.02,0.98),levels=lh)

#Compile list of conditional probability tables and create the network:

plist <- compileCPT(list(eh.o, bp.oil, oil.eh, rt.inf.eh, inf.oil.eh))

plist

summary(plist)

```
> summary(plist)
$eh
eh
 low high
 0.2  0.8
attr(,"class")
[1] "parray" "array"

$bp
        oil
bp         low high
  low     0.80  0.1
  normal 0.15  0.4
  high   0.05  0.5
attr(,"class")
[1] "parray" "array"

$oil
      eh
oil    low high
  low  0.9 0.05
  high 0.1 0.95
attr(,"class")
[1] "parray" "array"

$rt
, , eh = low

        inf
rt       low high
  low    0.6  0.2
  normal 0.3  0.2
  high   0.1  0.6

, , eh = high

        inf
```

```
rt        low high
  low     0.1 0.05
  normal 0.2 0.10
  high   0.7 0.85

attr(,"class")
[1] "parray" "array"

$inf
, , oil = low

      eh
inf     low high
  low  0.9  0.1
  high 0.1  0.9

, , oil = high

      eh
inf     low high
  low  0.2 0.02
  high 0.8 0.98

attr(,"class")
[1] "parray" "array"
```

3.3)

```
> net12 <- setEvidence(net1, evidence=list(rt="low", bp="high"))
> pEvidence( net12 )
[1] 0.005942
> net12 <- setEvidence(net1, evidence=list(rt="low", bp="high"))
> querygrain( net12, nodes=c("inf") )#Evidence can be entered in one of
these two equivalent forms:
$inf
inf
      low      high
0.2154157 0.7845843
```

net12 <- setEvidence(net1, evidence=list(rt="normal", bp="high"))

querygrain(net12, nodes=c("inf"))

```
$inf
inf
      low      high
0.1048185 0.8951815
```

$P(inf = high \mid bp = high, rt = normal)$ = 0.8951815

Q4)

4.1)

a) False – path exists
b) False - path via C-F-H un-blocked
c) True - Blocks at F.
d) False – path C-F-E-H is un-blocked
e) False – Path B-D-F-C-G un-blocked
f) True – C block all the paths
g) False - Path A-C-F-E-H unblocked

4.2)

```
library(igraph)
library(ggm)
#DAG
dag<- DAG(c~a, d~a+b, f~c+d+e, g~c, h~f+e)
drawGraph(dag, adjust = FALSE)

#d-separation
dSep(dag, first="c", second=c("g"), cond=NULL)
dSep(dag, first="c", second=c("h"), cond=c("e"))
dSep(dag, first="g", second=c("e"), cond=c("d"))
dSep(dag, first="c", second=c("h"), cond=c("f"))
dSep(dag, first="b", second=c("g"), cond=c("f"))
dSep(dag, first="b", second=c("g"), cond=c("d", "c", "e"))
dSep(dag, first="a", second=c("h"), cond=c("d", "f"))
```

```
> #d-separation
> dSep(dag, first="c", second=c("g"), cond=NULL)
[1] FALSE
> dSep(dag, first="c", second=c("h"), cond=c("e"))
[1] FALSE
> dSep(dag, first="g", second=c("e"), cond=c("d"))
[1] TRUE
> dSep(dag, first="c", second=c("h"), cond=c("f"))
[1] FALSE
> dSep(dag, first="b", second=c("g"), cond=c("f"))
[1] FALSE
> dSep(dag, first="b", second=c("g"), cond=c("d", "c", "e"))
[1] TRUE
> dSep(dag, first="a", second=c("h"), cond=c("d", "f"))
```

[1] FALSE

Q5)

5.1)

$$p(\mathrm{D}=1|\mathrm{A}=0)=\frac{p(D=1,A=0)}{p(A=0)}$$

$$=\frac{1}{\mathrm{P}(\mathrm{A}=0)}\sum_{B}\sum_{C}p(A=0)\,P(B|A=0)P(C|A=0,B)P(D=1|C)$$

$$=\frac{\mathrm{p}(\mathrm{A}=0)}{\mathrm{p}(\mathrm{A}=0)}\sum_{B,C}P(B|A=0)P(C|A=0,B)P(D=1|C)$$

$$=\sum_{B}P(B|A=0)\sum_{C}P(C|A=0,B)P(D=1|C)$$

$$=\sum_{B}P(B|A=0)\times[P(C=0|A=0,B)P(D=1|C=0)+P(C=1|A=0,B)P(D=1|C=1)]$$

$$=p(B=0|A=0)\times[P(C=0|A=0,B=0)P(D=1|C=0)+P(C=1|A=0,B=0)P(D=1|C=1)]+p(B=1|A=0)\times[P(C=0|A=0,B=1)P(D=1|C=0)+P(C=1|A=0,B=1)P(D=1|C=1)]$$

$$=0.2*[0.2*(1-\gamma)+0.8*0.7]+0.8*[0.4(1-\gamma)+0.6*(0.7)]$$

$$=0.2*(0.76-0.2\gamma)+0.8*(0.82-0.4\gamma)$$

$$=0.152-0.04\gamma+0.656-0.32\gamma$$

$$=0.808-0.36\gamma$$

$\mathrm{p}(\mathrm{D}=1|\mathrm{A}=0)$ *only depends on the* γ *values.*

5.2) $\mathrm{p}(\mathrm{D}=1|\mathrm{A}=0)=0.808-0.36\gamma=0.808-0.36*0.1=0.772$

Q6)

6.1) $P(F \mid \mathrm{A} = 1) = \frac{p(F, A=1)}{p(A=1)}$

6.2)

$$\mathrm{p(F, A)} = \sum_{B,C,D,E} p(A, B, C, D, E, F)$$

$$= \sum_{B,C,D,E} p(A)p(D)p(B|A)p(C|A, D)p(E|B, C)p(F|E, D)$$

$$= \sum_{B,C,D,E} f_0(A)f_1(D)f_2(A, B)f_3(A, C, D)f_4(B, C, E)f_5(D, E, F)$$

Observe A=1

$$\mathrm{p(F, A = 1)} = \sum_{B,C,D,E} f_1(D)f_6(B)f_7(C, D)f_4(B, C, E)f_5(D, E, F)$$

Eliminate B

$$\mathrm{p(F, A = 1)} = \sum_{C,D,E} f_1(D)f_7(C, D)f_5(D, E, F) \sum_{B} f_6(B)f_4(B, C, E)$$

$$= \sum_{C,D,E} f_1(D)f_7(C, D)f_5(D, E, F)f_8(C, E)$$

Eliminate C

$$\mathrm{p(F, A = 1)} = \sum_{D,E} f_1(D)f_5(D, E, F) \sum_{C} f_7(C, D)f_8(C, E) = \sum_{D,E} f_1(D)f_5(D, E, F)f_9(D, E)$$

Eliminate D

$$\mathrm{p(F, A = 1)} = \sum_{E} \sum_{D} f_1(D)f_5(D, E, F)f_9(D, E) = \sum_{E} f_{10}(E, F)$$

Eliminate E

$$\mathrm{p(F, A = 1)} = \sum_{E} f_{10}(E, F) = f_{11}(F)$$

Therefore, $\mathrm{p(F|A = 1)} = \frac{f_{11}(F)}{\sum_E f_{11}(F)}$

Q7) Two real world applications… [6 marks]