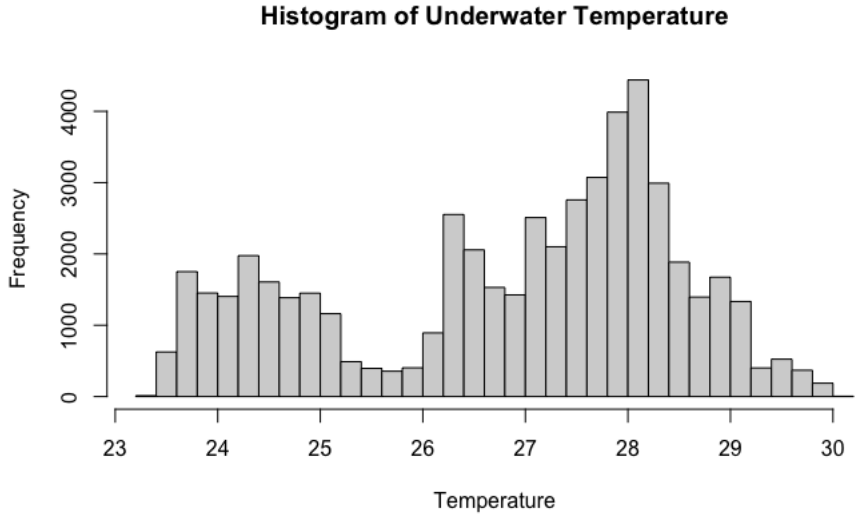
SIT743 Bayesian Learning and Graphical Models

Assignment - 2

Q1)

1.1)

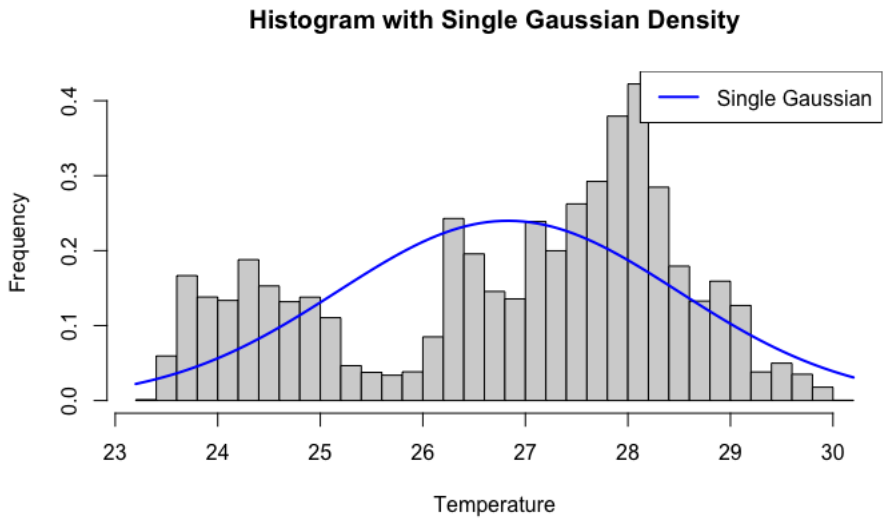
```
> #1.1
>
> # Read the CSV file
> water_temp_data <- read.csv("AIMSWaterTempData.csv", header = FALSE)
>
26.8302355358536
> # Plot the histogram
> hist(water_temp_data[,1], main = "Histogram of Underwater Temperature", xlab = "Temperature", ylab = "Frequency", breaks = 3
0)
>
```



2 modes have been observed from the histogram.

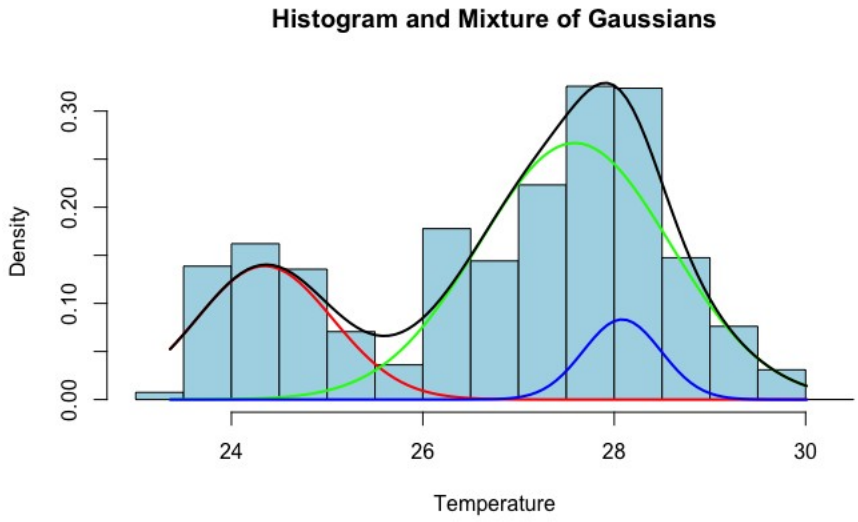
1.2)

```
> #1.2
>
> # Fit a single Gaussian model
> mu <- mean(water_temp_data[,1])
> sigma <- sd(water_temp_data[,1])
>
> # Plot the histogram and the single Gaussian density distribution
> hist(water_temp_data[,1], main = "Histogram with Single Gaussian Density", xlab = "Temperature", ylab = "Frequency", breaks
= 30, prob = TRUE)
> curve(dnorm(x, mean = mu, sd = sigma), col = "blue", lwd = 2, add = TRUE)
> legend("topright", legend = c("Single Gaussian"), col = c("blue"), lwd = 2)
>
> #Printing MLE estimates
>
> cat("Mean: ",mu)
Mean:  26.83024
> cat("\n")

> cat("Standard Deviation: ",sigma)
Standard Deviation:  1.66339
> cat("\n")
```



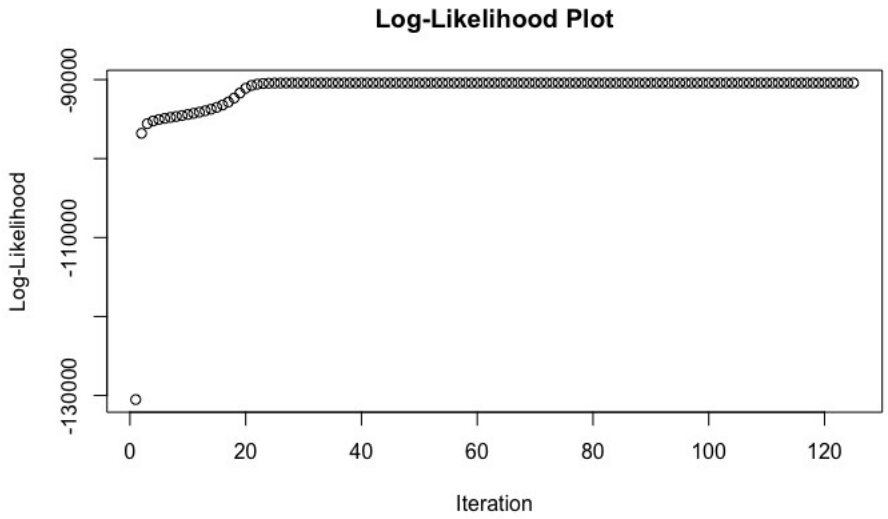
1.3)

```
>
> lines(x, density_component1, col = "red", lwd = 2)
> lines(x, density_component2, col = "green", lwd = 2)
> lines(x, density_component3, col = "blue", lwd = 2)
> lines(x, combined_density, col = "black", lwd = 2)
>
> # Print the mixing coefficients, means, and standard deviAions
>
> cat("Mixing Coefficients:\n")
Mixing Coefficients:
> print(mixing_coefficients)
[1] 0.24650569 0.66868355 0.08481075
>
> cat("\nMeans:\n")

Means:
> print(means)
[1] 24.34767 27.58662 28.08223
>
> cat("\nStandard Deviations:\n")

Standard Deviations:
> print(standard_deviations)
[1] 0.7080752 0.9996648 0.4060780
>
```



1.4)

```
> #1.4
>
> # Plot the log-likelihood values over iterations
> plot(mixture_model$all.loglik, main = "Log-Likelihood Plot", xlab = "Iteration", ylab = "Log-Likelihood")
>
```



From my analysis of the Log-Likelihood Plot over iterations for the mixture of the Gaussians model, I can make several observations. The plot shows how the log-likelihood, which measures the model's fit to the underwater temperature data, changes as the Expectation-Maximization (EM) algorithm iteratively updates the model parameters.

I noticed that the log-likelihood increases sharply in the early iterations, indicating rapid improvement in the model's fit. However, after around 40-60 iterations, the curve begins to flatten out, suggesting that the algorithm has converged to the optimal solution. This convergence is a good sign, as it means the model parameters have stabilized and further iterations would not yield significant enhancements. The final log-likelihood value of approximately -110,000 represents the best fit achieved by the mixture model with three components. While the specific magnitude of the log-likelihood is not as crucial as the overall pattern of convergence, the plot provides confidence that the EM algorithm has successfully found a stable and reliable solution for the given model configuration.

1.5)

When comparing the single Gaussian model (Q1.2) and the mixture of the Gaussian model (Q1.3), it's clear that the mixture model provides a better fit to the underwater temperature data. The key differences are:

- The single Gaussian model assumes an unimodal, symmetric distribution, which doesn't capture the multiple peaks and asymmetry in the data.
- The mixture of the Gaussians model, with three components, accurately represents the multimodal nature of the data, assigning each component to a distinct temperature range or cluster.
- The combined density plot of the mixture model aligns well with the histogram, capturing the shape and characteristics of the data more effectively.

In conclusion, the mixture of the Gaussian model is more appropriate and informative for this dataset, as it reveals the underlying structure and handles the multimodal nature of the underwater temperature distribution better than the single Gaussian model.

Q2)

Question-2.

2.1) Joint Distribution for the given network:

$$P(T,L,H,W,K,A,V,S) = P(T)\cdot P(L)\cdot P(H)\cdot P(W)\cdot P(K|T,L)\cdot P(A|L,H)\cdot P(V|W)\cdot P(S|K,A,V)$$

2.2) Minimum number of parameter of a node without parents = no. of states of node - 1.

For example, For T, there are 3 states,

$T \in \{$ less than 3000 (0), between 3000 & 20000 (1), and more than 20000 (2) $\}$

$P(T)$

$P(T=0) = ?$

$P(T=1) = ?$

$P(T=2) = 1 - P(T=0) - P(T=1)$

$\therefore$ 2 unknown parameters $\Rightarrow (3-1)$

similarly, For $(L) = 2-1 = 1$

For $(H) = 2-1 = 1$

For $(W) = 4-1 = 3$

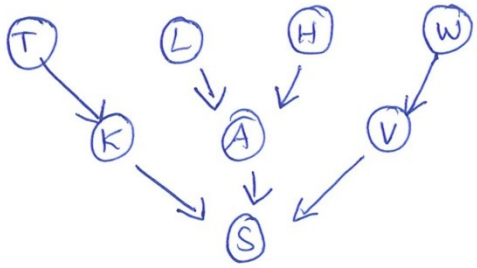
223296806 Ravi Shankar Jaganathan Senthil Kumar

2.3)

(c) Comparing both the results above, it can be seen that the number of parameters required 2591, is far greater than 65, when no independence is assumed or when all independencies are removed from the system.

2.4)

(a) After assuming that K is independent of L, given the Tonnage (T) $\Rightarrow P(K|L,T) = P(K|T)$ and so the connection between K and L can be removed from the initial network. ∴ The new Bayesian network is:



2.4)

(b) Computing the minimum number of parameters for the new network:

For (T) = 3-1 = 2

For (L) = 2-1 = 1

For (H) = 2-1 = 1.

For (W) = $4-1 = 3$

For (K) = $3 \times (2-1) = 3$

For (A) = $2 \times 2 \times (3-1) = 8$

For (V) = $4 \times (3-1) = 8$

For (S) = $3 \times 2 \times 3 \times (3-1) = 36$

$\therefore$ Total = $2 + 1 + 1 + 3 + 3 + 8 + 8 + 36$

$= 62.$

Comparing with the previous results, it can be seen that when a new independence factor was introduced, the minimum number of parameter required decreases, here it being a reduction of 3 parameters (65-62)

2.5)

(a) Is (H) independent of (W) given (K), (S) & (A).

Paths:- (1) (H) → (A) → (S) → (V) → (W)

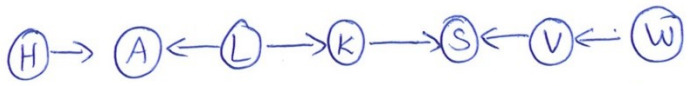
Here, (S) & (A) are observed

$\therefore$ At (A), as Head to tail and observed

→ Blocked.

223296806 Ravi Shankar Jaganathan Senthil Kumar

(5)

(2) 

Here, (A), (K) & (S) are observed.

∴ At (K), as Head to tail and observed

⇒ Blocked.

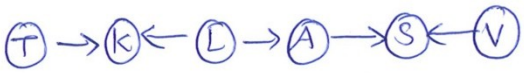
∴ All paths are blocked and (H) is independent of (W) given (K), (S) and (A)

(b) Is $T \perp V$?

Paths 1) 

At (S), Head to Head and unobserved

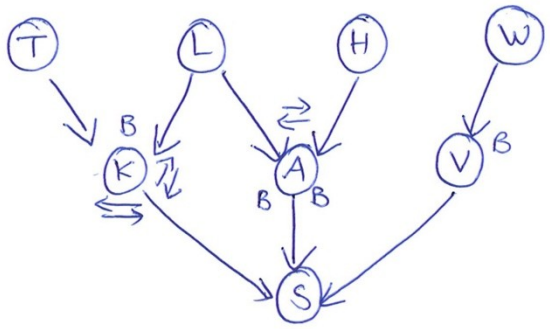
⇒ Blocked.

2) 

At (K) and (S), Head to Head and unobserved ⇒ Blocked.

∴ All paths are blocked and $T \perp V$ is true.

2.6)



Here A & V are observed ; all paths are checked and, B denotes blocked and $\rightleftarrows$ denotes flowing.

∴ According to the diagram

At (K) ⇒ blocked, as head to head and unobserved, so $T \perp L$.

At (A) ⇒ path from (S) to (H) is blocked, as tail to head and observed,

So $T \perp H$

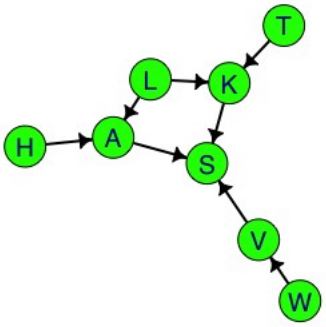
At (V) ⇒ path from (S) to (W) is blocked, as tail to Head and observed, so $T \perp W$.

∴ T is conditionally independent of $\{L, H, W\}$ given A and V.

2.7)

Bayesian Network Plot

```
> #2.7
>
> #BiocManager::install("graph")
> #BiocManager::install("ggm")
> library(graph)
> library(ggm)
>
> # Define the Bayesian network structure
> dag <- DAG(K ~ T + L, A ~ L + H, V ~ W, S ~ K + A + V)
>
> # Plot the Bayesian network
> plotGraph(dag, nodesize = 20, tcltk = FALSE, vc = "green")
>
```



2.7) - (A)

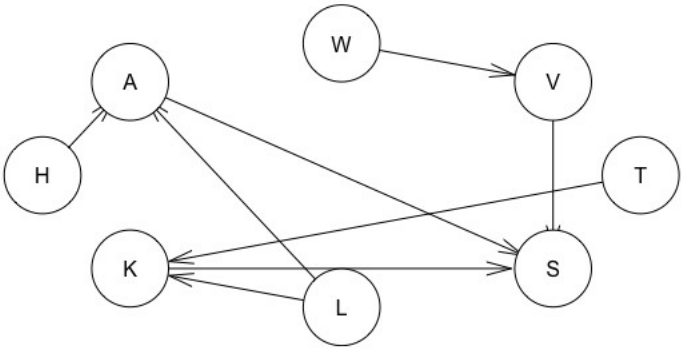
```
> # Q2.7 (a)
> # W ⊥ K | S
> dSep(dag, first = "W", second = "K", cond = "S") # False
[1] FALSE
>
```

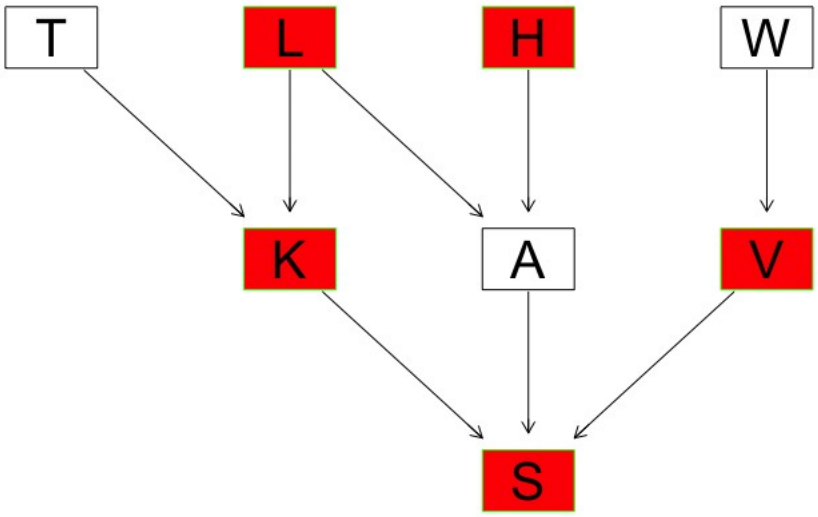
2.7) (B)

```
> # Q2.7 (b)
0.82258064516129 K,A}
> dSep(dag, first = "H", second = "V", cond = c("K", "A")) # True
[1] TRUE
>
```

2.8) (A)

```
> # Q2.8 (a)
> # Markov blanket of A (Accident type)
> library(bnlearn)
> if (!requireNamespace("BiocManager", quietly = TRUE))
+   install.packages("BiocManager")
> #BiocManager::install("Rgraphviz")
>
> model_net = model2network("[T][L][H][W][K|T:L][A|L:H][V|W][S|K:A:V]")
> plot(model_net)
>
> # Markov blanket
> mb(model_net, "A")
[1] "H" "K" "L" "S" "V"
>
> # Plot the Markov blanket of A in the network
> graphviz.plot(model_net, highlight = list(nodes = mb(model_net, "A"), col = "green", fill = "red"))
>
```





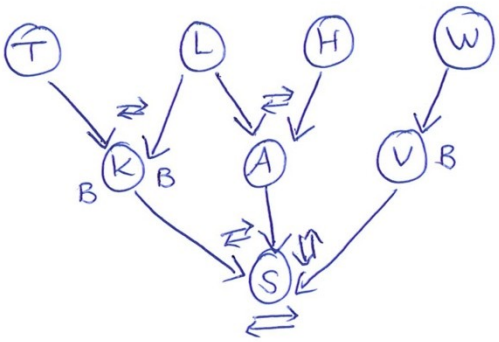
2.8)

(a) Markov blanket of a node =

parents + children + parents of it's children.

∴ Markov blanket of A = {L, H, S, K, V}

(b)



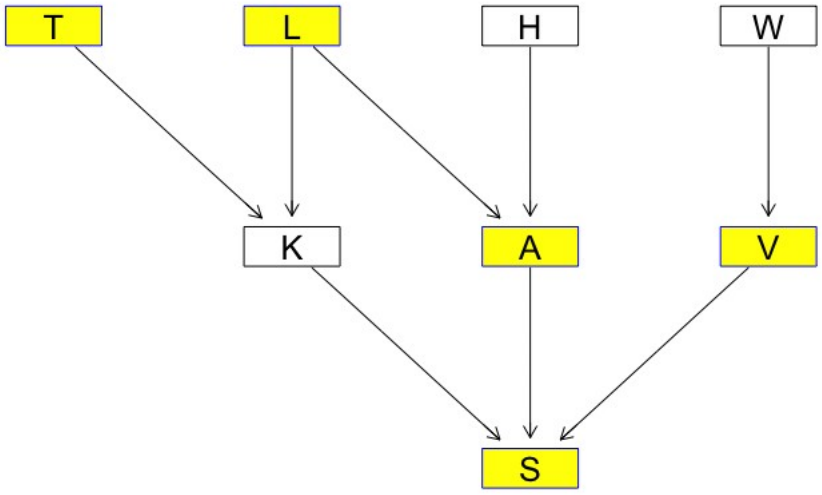
Here we check if A is independent of T or W. The path from A to T is blocked at (L); tail to tail and observed and (K); tail to head and observed

The path from A to W is blocked at (V); as tail to head and observed.

∴ A is independent of T and W, given it's Markov blanket.

2.8) - (C)

```
> # Q2.8 (c)
> # Find the Markov blanket of K (Kilowatt Power) and plot it
> mb(model_net, "K")
[1] "A" "L" "S" "T" "V"
> graphviz.plot(model_net, highlight = list(nodes = mb(model_net, "K"), col = "blue", fill = "yellow"))
>
```



2.9)

(a) $$P(S|W,A,V,K) = \frac{P(S,W,A,V,K)}{P(W,A,V,K)}$$

$$= \frac{P(S,W,A,V,K)}{\sum_S P(S,W,A,V,K)}$$

$$P(S,W,A,V,K) = \sum_{T,L,H} P(S,W,A,V,K,T,L,H)$$

$$= \sum_{T,L,H} P(T)\cdot P(L)\cdot P(H)\cdot P(W)\cdot P(K|T,L)\cdot P(A|L,H)\cdot P(V|W)\cdot P(S|K,A,V)$$

$$= \sum_{T,L,H} f_0(T) f_1(L) f_2(H) f_3(W) f_4(K,T,L) f_5(A,L,H)\cdot f_6(V,W)\cdot f_7(S,K,A,V)$$

observing W = Fog, A = Catastrophic, V = High and K = less than 3000. P(S|W = Fog, A = Catastrophic, V = High, K = less than 3000)

$$= \sum_{T,L,H} f_0(T) f_1(L) f_2(H) f_8(T,L) f_9(L,H) f_{10}(S)$$

Eliminating T

(9)

$$= \sum_{L,H} f_1(L)\, f_2(H)\, f_9(L,H)\, f_{10}(S) \cdot \sum_{T} f_0(T) \cdot f_8(T,L)$$

$$= \sum_{L,H} f_1(L)\, f_2(H)\, f_9(L,H)\, f_{10}(S) \cdot f_{11}(L)$$

Eliminating L

$$= \sum_{H} f_2(H) \cdot f_{10}(S) \cdot \sum_{L} f_1(L)\, f_9(L,H) \cdot f_{11}(L)$$

$$= \sum_{H} f_2(H)\, f_{10}(S)\, f_{12}(H)$$

Eliminating H

$$= f_{10}(S) \sum_{H} f_2(H)\, f_{12}(H)$$

$$= f_{10}(S)$$

$\therefore P(S \mid W = \text{Fog}, A = \text{Catastrophic}, V = \text{High}, K = \text{less than 3000})$

$$= \frac{f_{10}(S)}{\sum_{S} f_{10}(S)}$$

223296806 Ravi Shankar Jaganathan Senthil Kumar

2.9)

(b) Treewidth of a network, is the maximum number of variables in a factor created by summing out a variable, given the elimination ordering.

∴ While eliminating $T_1 = 1$.

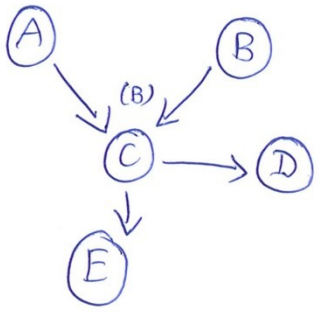
While eliminating $L_1 = 1$.

While eliminating $H_1 = 0$.

∴ Tree-width = 1.

Question - 3

3.1)



Here, it can be seen that there is no path from (A) to (B) as at (C), it is head to head and unobserved, hence, blocked. Therefore $A \perp B$

3.2) $$P(E=0 \mid A=0, B=1) = \frac{P(E=0, A=0, B=1)}{P(A=0, B=1)}$$

$$= \frac{P(E=0, A=0, B=1)}{\sum_{E} P(E, A=0, B=1)}$$

Converting into factors,

$$P(A,B,E) = \sum_{C,D} P(A) \cdot P(B) \cdot P(C \mid A,B) \cdot P(D \mid C) \cdot P(E \mid C)$$

$$= \sum_{C,D} \phi_0(A) \cdot \phi_1(B) \cdot \phi_2(C,A,B) \, \phi_3(D,C) \, \phi_4(E,C)$$

$P(A) \rightarrow \phi_0(A) \rightarrow$

A	$\phi_0(A)$
0	α
1	$1-\alpha$

223296806 Ravi Shankar Jaganathan Senthil Kumar

$P(B)$ ↓ $f_1(B)$

B	$f_1(B)$
0	γ
1	$1-\gamma$

$P(C|A,B)$ ↓ $f_2(C,A,B)$

A	B	C	$f_2(A,B,C)$
0	0	0	0.1
0	0	1	0.4
0	0	2	0.5
0	1	0	0.5
0	1	1	0.2
0	1	2	0.3
1	0	0	0.2
1	0	1	λ
1	0	2	$(0.8-\lambda)$
1	1	0	0.1
1	1	1	0.1
1	1	2	0.8

$P(D|C)$ ↓ $f_3(D,C)$

C	D	$f_3(D,C)$
0	0	0.7
0	1	0.3
1	0	σ
1	1	$(1-\sigma)$
2	0	0.4
2	1	0.6

$P(E|C)$ ↓ $f_4(E,C)$

C	E	$f_4(C,E)$
0	0	0.4
0	1	0.6
1	0	β
1	1	$(1-\beta)$
2	0	0.8
2	1	0.2

Observe, $A=0$, $B=1$.

Then $P(A,B,C) = \sum_{C,D} f_0(A) \cdot f_1(B) \cdot f_2(A,B,C) \cdot f_3(C,D) \cdot f_4(C,E)$

$\{ f_2(A,B,C) \downarrow f_5(C) \}$ Becomes $\Rightarrow \sum_{C,D} f_5(C) \cdot f_3(C,D) \cdot f_4(C,E)$

After observing,

$f_2(A,B,C)$ table $\Rightarrow f_5(C)$

$(A=0, B=1)$

C	$f_5(C)$
0	0.5
1	0.2
2	0.3

Eliminating D,

$$\Rightarrow \sum_{C} f_5(C) \cdot f_4(C,E) \sum_{D} f_3(C,D)$$

$$= \sum_{C} f_5(C) \cdot f_4(C,E) \cdot f_6(C)$$

$\sum_{D} f_3(C,D) \rightarrow f_6(C) \Rightarrow f_6(C)$

C	$f_6(C)$
0	$0.7+0.3=1.$
1	$5+1-5=1$
2	$0.6+0.4=1.$

$\therefore$ Multiply $f_6(C) \cdot f_5(C) \Rightarrow f_7(C)$

C	$f_7(C)$
0	0.5
1	0.2
2	0.3

(4)

$$= \sum_C f_7(C) \cdot f_4(C,E)$$

$\Rightarrow$ Multiplying $f_7(C) \cdot f_4(C,E) = f_8(C,E)$

C	$f_7(C)$
0	0.5
1	0.2
2	0.3

$\times$

C	E	$f_4(C,E)$
0	0	0.4
0	1	0.6
1	0	β
1	1	$1-\beta$
2	0	0.8
2	1	0.2

$\Rightarrow$

C	E	$f_8(C,E)$
0	0	$0.4 \times 0.5 = 0.2$
0	1	$0.6 \times 0.5 = 0.3$
1	0	0.2β
1	1	$0.2 - 0.2\beta$
2	0	$0.8 \times 0.3 = 0.24$
2	1	$0.2 \times 0.3 = 0.06$

Finally eliminating C $\Rightarrow \sum_C f_8(C,E)$

$= f_9(E)$

$\therefore \sum_C f_8(C,E) \Rightarrow$

E	$f_9(E)$
0	$0.2+0.2\beta+0.24 = 0.44+0.2\beta$
1	$0.3+0.2-0.2\beta+0.06 = 0.56-0.2\beta$

$\Rightarrow P(E=0, A=0, B=1) = f_9(E)$

$\therefore P(E=0 \mid A=0, B=1) = \dfrac{f_9(E)}{\sum_E f_9(E)}$

223296806 Ravi Shankar Jaganathan Senthil Kumar

$$= \frac{f_9(E=0)}{f_9(E=0) + f_9(E=1)}$$

$$= \frac{0.44 + 0.2\beta}{0.44 + 0.2\beta + 0.56 - 0.2\beta}$$

$$= 0.44 + 0.2\beta$$

3.3) Maximum likelihood estimate is the relative frequency.

$$\Rightarrow \alpha = P(A=0) = \frac{7}{30} = 0.23$$

$$\beta = P(E=0 \mid C=1) = \frac{P(E=0, C=1)}{P(C=1)}$$

$$= \frac{2}{6} = \frac{1}{3} = 0.33$$

3.4) $P(E=0 \mid A=0, B=1)$

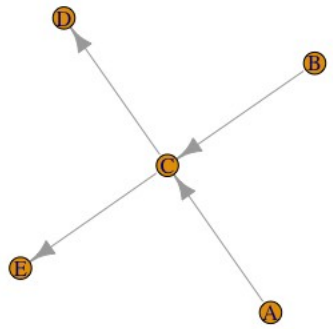
Given $\beta = 0.33$

Substituting in the previous equation

$$\Rightarrow 0.44 + 0.2\,(0.33) = 0.506$$

3.5) (A)

```
> #3.5 (a)
>
> # BiocManager::install(c("gRain", "RBGL", "gRbase"))
> # BiocManager::install(c("Rgraphviz"))
>
> library("Rgraphviz")
>
num [1:3] 24.4 27.9 26.4
> #install.packages("gRain")
> library(gRbase)
> library(gRain)
>
> #Assigning alpha=0.3, beta=0.6, lambda=0.5, gamma=0.2, sigma=0.3
>
> alpha <- 0.3
> beta <- 0.6
> lambda <- 0.5
> gamma <- 0.2
> sigma <- 0.3
>
> # CPT for A
> A_cpt <- cptable(~ A, values = c(alpha, 1 - alpha), levels = c("0", "1"))
>
> # CPT for B
> B_cpt <- cptable(~ B, values = c(gamma, 1 - gamma), levels = c("0", "1"))
>
> # CPT for C
> C_cpt <- cptable(~ C | A:B, values = c(0.1, 0.4, 0.5, 0.5, 0.2, 0.3, 0.2, lambda, (0.8 - lambda), 0.1, 0.1, 0.8), levels = c
("0", "1", "2"))
>
> # CPT for D
> D_cpt <- cptable(~ D | C, values = c(0.7, 0.3, sigma, (1 - sigma), 0.4, 0.6), levels = c("0", "1"))
>
> # CPT for E
> E_cpt <- cptable(~ E | C, values = c(0.4, 0.6, beta, (1 - beta), 0.8, 0.2), levels = c("0", "1"))
>
> # Define the network structure
> network_structure <- grain(compileCPT(list(A_cpt, B_cpt, C_cpt, D_cpt, E_cpt)))
>
> # Plot the belief network
> plot(network_structure$dag)
>
```



3.5) (B)

```
> #Q3.5 (B)
>
> compiled_nodes <- compileCPT(list(A_cpt, B_cpt, C_cpt, D_cpt, E_cpt))
>
> # Probability table for A
> print(compiled_nodes$A)
A
  0   1
0.3 0.7
>
> # Probability table for B
> print(compiled_nodes$B)
B
  0   1
0.2 0.8
>
> # Probability table for C
> print(compiled_nodes$C)
, , B = 0

   A
C     0   1
  0 0.1 0.5
  1 0.4 0.2
  2 0.5 0.3

, , B = 1

   A
C     0   1
  0 0.2 0.1
  1 0.5 0.1
  2 0.3 0.8

>
> # Probability table for D
> print(compiled_nodes$D)
   C
D     0   1   2
  0 0.7 0.3 0.4
  1 0.3 0.7 0.6
>
> # Probability table for E
```

```
> # Probability table for E
> print(compiled_nodes$E)
   C
E     0   1   2
  0 0.4 0.6 0.8
  1 0.6 0.4 0.2
>
```

3.5) (C)

```
> #Q3.5 (C)
>
> #i)
> # Marginal distribution of C
> querygrain(network_structure, nodes = "C")
$C
C
    0     1     2
0.180 0.228 0.592

>
> #ii)
> # Joint distribution of A, B, and C
> querygrain(network_structure, nodes = c("A", "B", "C"), type="joint")
, , B = 0

   C
A       0     1     2
  0 0.006 0.024 0.030
  1 0.070 0.028 0.042

, , B = 1

   C
A       0     1     2
  0 0.048 0.120 0.072
  1 0.056 0.056 0.448

>
> #iii)
> # Set the evidence for A=0, D=1, and E=0
> network_with_evidence <- setEvidence(network_structure, nodes = c("A", "D", "E"), states = c("0", "1", "0"))
>
> # Query the network for P(C=2 | A=0, D=1, E=0)
> querygrain(network_with_evidence, nodes = "C")
$C
C
         0          1          2
0.05590062 0.52173913 0.42236025

>
```

Q4)

```
> library(bnlearn)
>
> # Load the alarm dataset
> data(alarm)
> summary(alarm)
     CVP           PCWP          HIST           TPR            BP            CO           HRBP
 HIGH  : 3159   HIGH  : 4247   FALSE:18909   HIGH  :5967   HIGH  :7262   HIGH  :11000   HIGH  :12968
 LOW   : 2285   LOW   : 2278   TRUE : 1091   LOW   :6135   LOW   :7875   LOW   : 3624   LOW   : 5792
 NORMAL:14556   NORMAL:13475                 NORMAL:7898   NORMAL:4863   NORMAL: 5376   NORMAL: 1240

     HREK           HRSA           PAP           SAO2          FIO2          PRSS           ECO2
 HIGH  :12561   HIGH  :12587   HIGH  : 1151   HIGH  : 3422   LOW   : 1011   HIGH  :10270   HIGH  :  707
 LOW   : 5579   LOW   : 5580   LOW   : 1012   LOW   :15926   NORMAL:18989   LOW   : 5093   LOW   : 5922
 NORMAL: 1860   NORMAL: 1833   NORMAL:17837   NORMAL:  652                  NORMAL: 4105   NORMAL:  492
                                                                            ZERO  :  532   ZERO  :12879
     MINV           MVS           HYP           LVF           APL           ANES          PMB
 HIGH  : 3781   HIGH  : 1035   FALSE:15955   FALSE:19010   FALSE:19798   FALSE:17907   FALSE:19795
 LOW   : 1262   LOW   : 1041   TRUE : 4045   TRUE :  990   TRUE :  202   TRUE : 2093   TRUE :  205
 NORMAL:  644   NORMAL:17924
 ZERO  :14313
        INT           KINK          DISC           LVV           STKV          CCHL          ERLO
 ESOPHAGEAL:  606   FALSE:19273   FALSE:18027   HIGH  : 4319   HIGH  :  833   HIGH  :15126   FALSE:18998
 NORMAL    :18396   TRUE :  727   TRUE : 1973   LOW   : 1765   LOW   : 3574   NORMAL: 4874   TRUE : 1002
 ONESIDED  :  998                               NORMAL:13916   NORMAL:15593

      HR           ERCA           SHNT           PVS           ACO2          VALV          VLNG
 HIGH  :13774   FALSE:18012   HIGH  : 2060   HIGH  : 3539   HIGH  :  196   HIGH  : 3759   HIGH  :  552
 LOW   :  386   TRUE : 1988   NORMAL:17940   LOW   :16019   LOW   :17211   LOW   : 1532   LOW   : 4340
 NORMAL: 5840                                NORMAL:  442   NORMAL: 2593   NORMAL:  825   NORMAL:  234
                                                                           ZERO  :13884   ZERO  :14874
     VTUB           VMCH
 HIGH  : 1171   HIGH  : 1165
 LOW   :14667   LOW   : 1175
 NORMAL:  329   NORMAL:16682
 ZERO  : 3833   ZERO  :  978
>
```

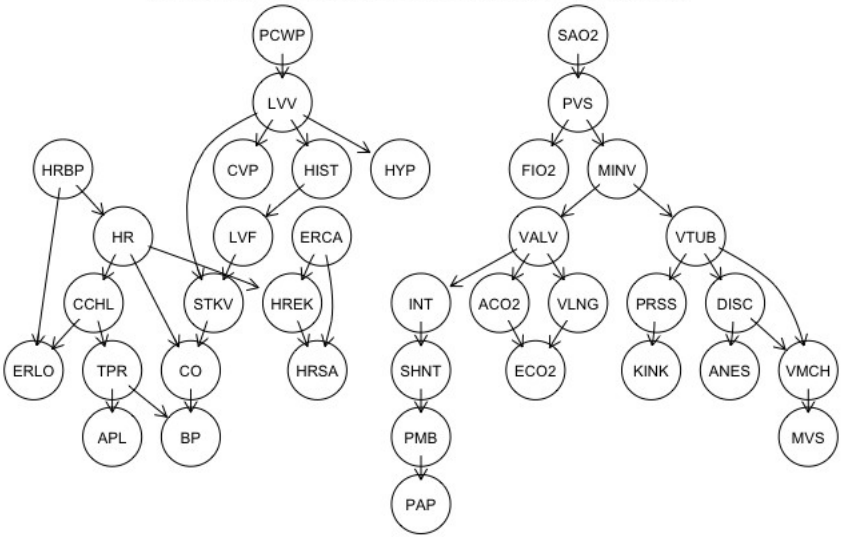
```
>
> # Create the true network structure
> true_network_structure <- paste0("[HIST|LVF][CVP|LVV][PCWP|LVV][HYP][LVV|HYP:LVF][LVF]",
+                                  "[STKV|HYP:LVF][ERLO][HRBP|ERLO:HR][HREK|ERCA:HR][ERCA][HRSA|ERCA:HR][ANES]",
+                                  "[APL][TPR|APL][ECO2|ACO2:VLNG][KINK][MINV|INT:VLNG][FIO2][PVS|FIO2:VALV]",
+                                  "[SAO2|PVS:SHNT][PAP|PMB][PMB][SHNT|INT:PMB][INT][PRSS|INT:KINK:VTUB][DISC]",
+                                  "[MVS][VMCH|MVS][VTUB|DISC:VMCH][VLNG|INT:KINK:VTUB][VALV|INT:VLNG]",
+                                  "[ACO2|VALV][CCHL|ACO2:ANES:SAO2:TPR][HR|CCHL][CO|HR:STKV][BP|CO:TPR]")
>
> true_dag <- model2network(true_network_structure)
> par(mfrow = c(1, 1))
> graphviz.plot(true_dag, shape = "circle")
>
```

4.1)

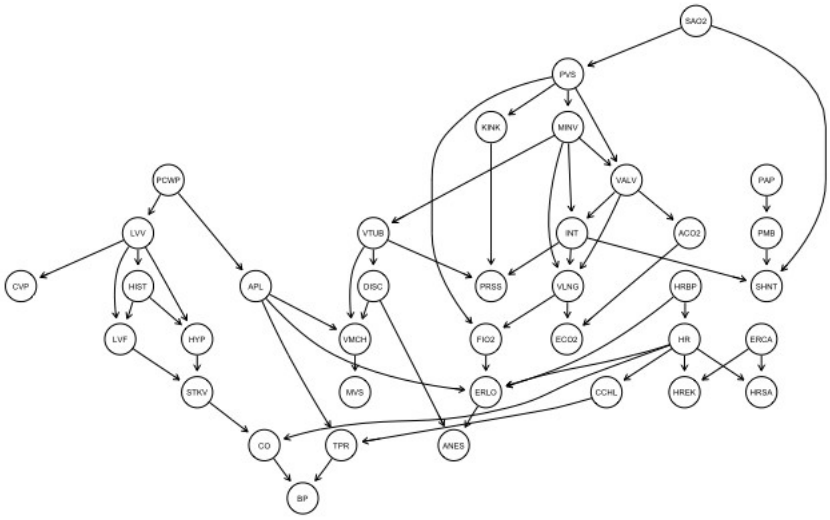
```
> # Q4.1 Learn Bayesian network structures using hill-climbing algorithm with BIC and BDe scores
> data_sizes <- c(500, 5000, 10000)
>
> cat("Question 4.1\n")
Question 4.1
>
> for (size in data_sizes) {
+   alarm_data <- alarm[1:size, ]
+
+   # Learn network structure using hill-climbing with BIC score
+   hc_bic_model <- hc(alarm_data, score = "bic")
+   bic_score_value <- score(hc_bic_model, alarm_data, type = "bic")
+   cat(sprintf("BIC score for sample size %d: %f\n", size, bic_score_value))
+   graphviz.plot(hc_bic_model, main = paste("Hill-climbing with BIC score (Sample size:", size, ")"), shape = "circle")
+
+   # Learn network structure using hill-climbing with BDe score
+   hc_bde_model <- hc(alarm_data, score = "bde")
+   bde_score_value <- score(hc_bde_model, alarm_data, type = "bde")
+   cat(sprintf("BDe score for sample size %d: %f\n", size, bde_score_value))
+   graphviz.plot(hc_bde_model, main = paste("Hill-climbing with BDe score (Sample size:", size, ")"), shape = "circle")
+ }
BIC score for sample size 500: -6299.547648
BDe score for sample size 500: -5789.679290
BIC score for sample size 5000: -56013.150674
BDe score for sample size 5000: -54803.771999
BIC score for sample size 10000: -111246.476683
BDe score for sample size 10000: -109390.674884
>
> cat("\n")

>
```

```
Question 4.1
BIC score for sample size 500: -6299.547648
BDe score for sample size 500: -5789.679290
BIC score for sample size 5000: -56013.150674
BDe score for sample size 5000: -54803.771999
BIC score for sample size 10000: -111246.476683
BDe score for sample size 10000: -109390.674884
```

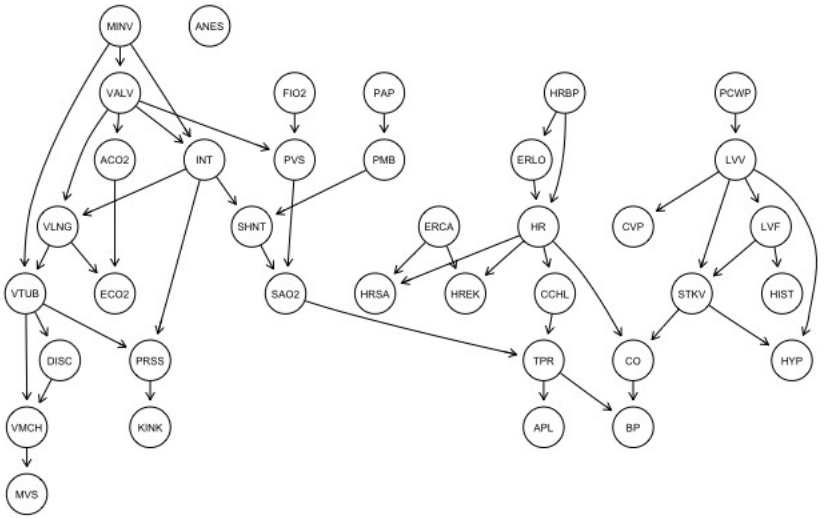
Hill-climbing with BIC score (Sample size: 500)



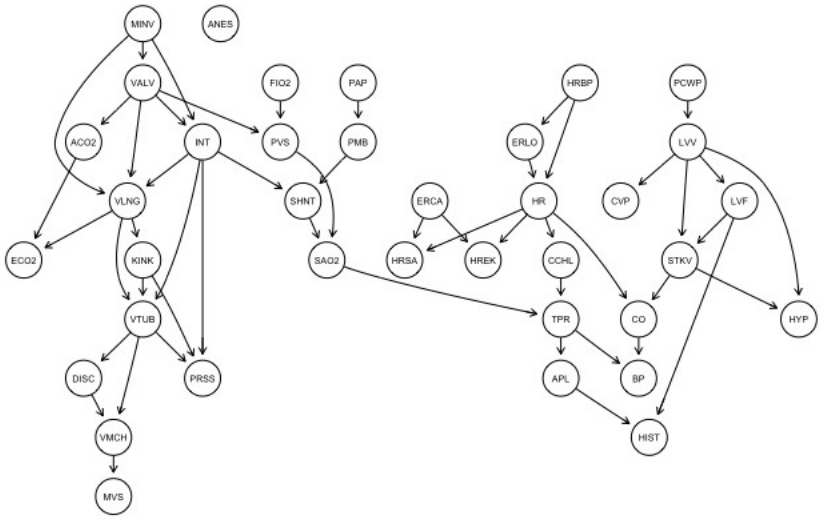
Hill-climbing with BDe score (Sample size: 500)



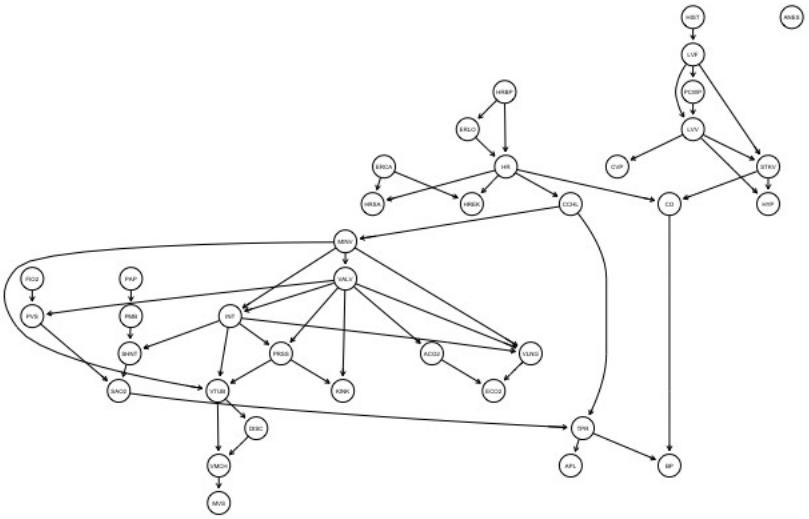
Hill-climbing with BIC score (Sample size: 5000)

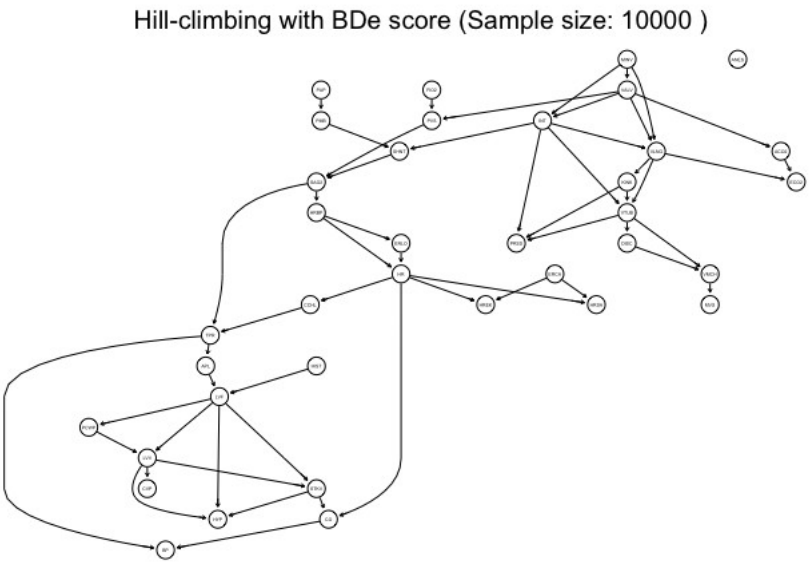


Hill-climbing with BDe score (Sample size: 5000)



Hill-climbing with BIC score (Sample size: 10000)





4.2)

Based on the results I obtained for question 4.1 and the image, I can discuss how the BIC score compares with the BDe score for different sample sizes in terms of structure and score of the learned network.

Regarding the structure, I observed that as the sample size increases (500, 5000, and 10000), the network structures learned using the hill-climbing algorithm with both BIC and BDe scores become more complex and denser. I noticed an increasing number of edges and connections between the nodes in the learned networks as the sample size grew.

When I compared the structures learned using BIC and BDe scores for the same sample size, I found that they were relatively similar. However, I spotted some differences in the specific connections and orientations of the edges between the nodes. This led me to believe that both scoring methods tend to converge towards similar structures as the sample size increases, but they may capture slightly different dependencies and relationships among the variables.

Moving on to the scores, I observed from the output that both BIC and BDe scores decrease (become more negative) as the sample size increases. This aligns with my understanding that both scores penalize model complexity, and as the sample size grows, the learned networks become more complex, resulting in lower scores.

When I compared the BIC and BDe scores for the same sample size, I noticed that the BDe scores were consistently higher (less negative) than the

BIC scores. This suggests to me that the BDe score tends to favour slightly more complex networks compared to the BIC score. I also observed that the difference between the scores becomes more pronounced as the sample size increases.

For instance, at a sample size of 500, I found the BIC score to be -6299.55, and the BDe score to be -5789.68, showing a difference of around 509.87. However, at a sample size of 10000, the BIC score is -111246.48, and the BDe score is -109390.67, with a difference of approximately 1855.81. This observation suggests to me that the BDe score is more lenient towards complex networks, especially as the sample size grows.

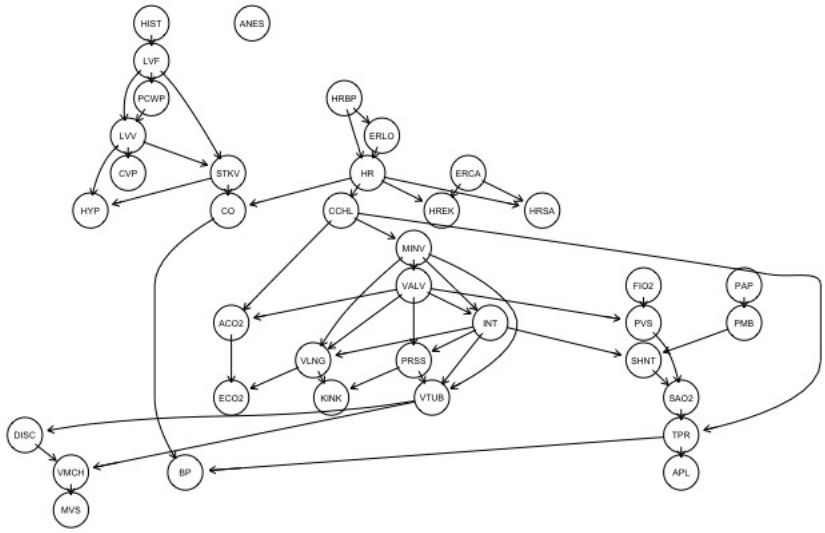
In conclusion, based on my analysis, both BIC and BDe scores lead to increasingly complex network structures as the sample size increases. The learned structures are relatively similar between the two scoring methods, but there may be some differences in the specific connections. I found that the BDe score tends to favour slightly more complex networks compared to the BIC score, and this difference becomes more pronounced with larger sample sizes. In my opinion, the choice between BIC and BDe scores may depend on the specific problem domain and the desired balance between model complexity and goodness of fit.

4.3) (A)

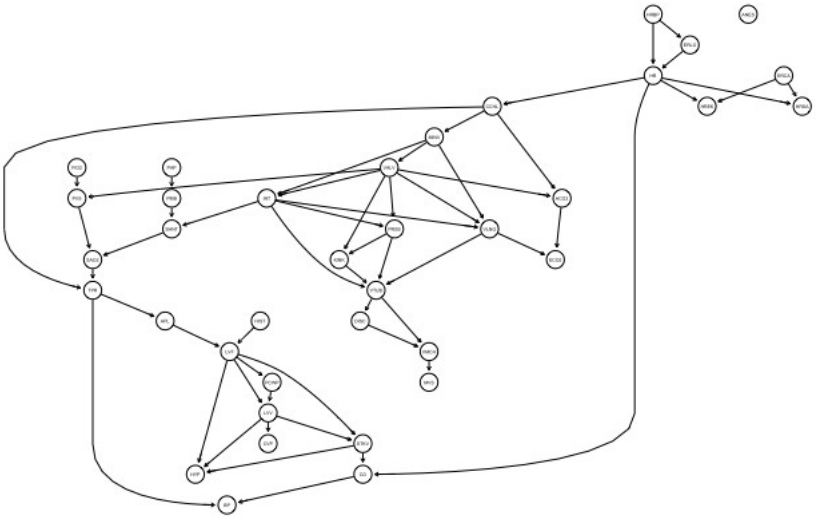
```
> # Q4.3 (a) Learn Bayesian network structures using the full dataset with BIC and BDe scores
> cat("Question 4.3 (a)\n")
Question 4.3 (a)
>
> hc_bic_full_model <- hc(alarm, score = "bic")
> bic_full_score <- score(hc_bic_full_model, alarm, type = "bic")
> cat(sprintf("BIC score for full dataset: %f\n", bic_full_score))
BIC score for full dataset: -220761.687713
> graphviz.plot(hc_bic_full_model, main = "Hill-climbing with BIC score (Full dataset)", shape = "circle")
>
> hc_bde_full_model <- hc(alarm, score = "bde")
> bde_full_score <- score(hc_bde_full_model, alarm, type = "bde")
> cat(sprintf("BDe score for full dataset: %f\n", bde_full_score))
BDe score for full dataset: -218700.186025
> graphviz.plot(hc_bde_full_model, main = "Hill-climbing with BDe score (Full dataset)", shape = "circle")
>
> cat("\n")

>
```

```
Question 4.3 (a)
BIC score for full dataset: -220761.687713
BDe score for full dataset: -218700.186025
```

Hill-climbing with BIC score (Full dataset)



Hill-climbing with BDe score (Full dataset)

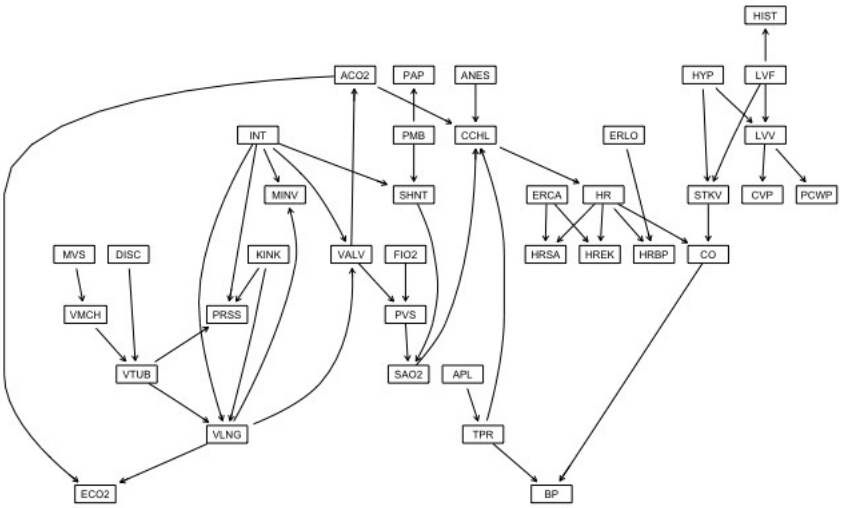


4.3) (B)

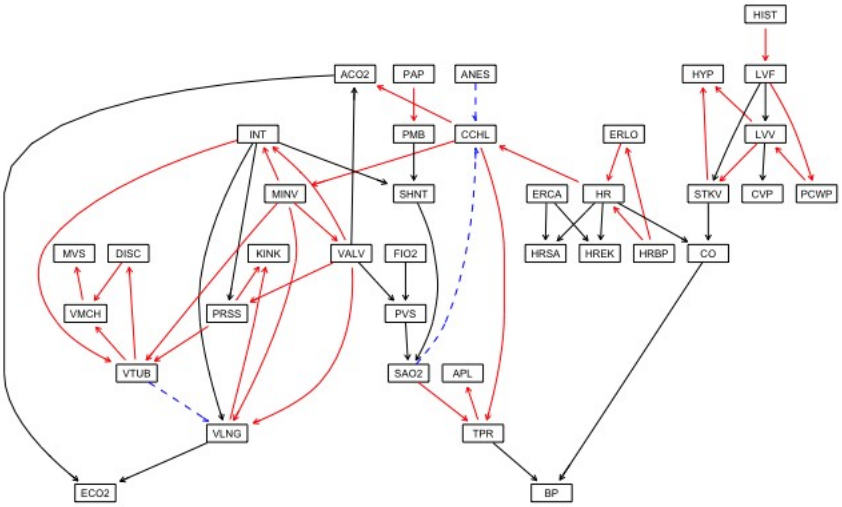
```
> # Q4.3 (b) Compare the learned networks with the true network structure
> cat("Question 4.3 (b)\n")
Question 4.3 (b)
>
> bic_comparison <- compare(true_dag, hc_bic_full_model)
> bde_comparison <- compare(true_dag, hc_bde_full_model)
>
> cat("Comparison of BIC network with true network:\n")
Comparison of BIC network with true network:
> print(bic_comparison)
$tp
[1] 22

$fp
[1] 31

$fn
[1] 24

> graphviz.compare(true_dag, hc_bic_full_model, main = c("True Network", "BIC Network"))
>
> cat("Comparison of BDe network with true network:\n")
Comparison of BDe network with true network:
> print(bde_comparison)
$tp
[1] 22

$fp
[1] 34

$fn
[1] 24

> graphviz.compare(true_dag, hc_bde_full_model, main = c("True Network", "BDe Network"))
>
> cat("\n")

>
```

```
Question 4.3 (b)
Comparison of BIC network with true network:
$tp
[1] 22

$fp
[1] 31

$fn
[1] 24

Comparison of BDe network with true network:
$tp
[1] 22

$fp
[1] 34

$fn
[1] 24
```

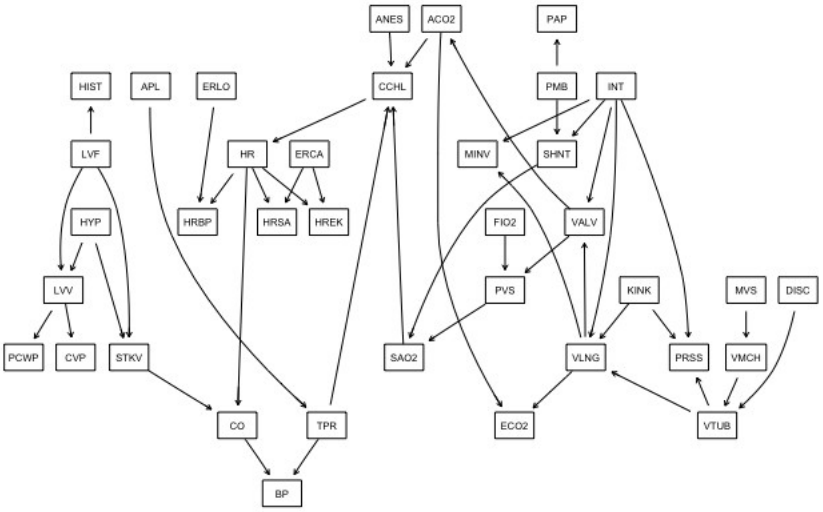
True Network

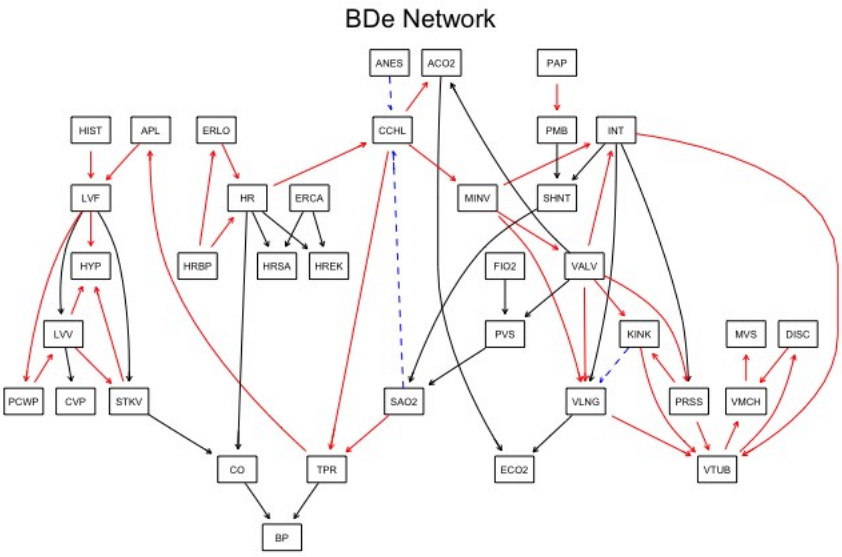


BIC Network



True Network




BDe Network
ANES
ACO2
PAP
HIST
APL
ERLO
CCHL
PMB
INT
LVF
HR
ERCA
MINV
SHNT
HYP
HRBP
HRSA
HREK
FIO2
VALV
LVV
PVS
KINK
MVS
DISC
PCWP
CVP
STKV
SAO2
VLNG
PRSS
VMCH
CO
TPR
ECO2
VTUB
BP

4.3) (C)

```
> # Q4.3 (c) Fit the data to the BIC network and show CPD table for "HR"
> cat("Question 4.3 (c)\n")
Question 4.3 (c)
>
> bic_fitted_model <- bn.fit(hc_bic_full_model, alarm)
> hr_cpd_table <- bic_fitted_model$HR
> cat("CPD table for 'HR' variable:\n")
CPD table for 'HR' variable:
> print(hr_cpd_table)

  Parameters of node HR (multinomial distribution)

Conditional probability table:

, , ERLO = FALSE

        HRBP
HR               HIGH          LOW       NORMAL
  HIGH   0.9950306701 0.0226395226 0.3444180523
  LOW    0.0004658747 0.0242190242 0.5415676960
  NORMAL 0.0045034552 0.9531414531 0.1140142518

, , ERLO = TRUE

        HRBP
HR               HIGH          LOW       NORMAL
  HIGH   0.0224719101 0.0744680851 0.8253968254
  LOW    0.0000000000 0.1489361702 0.0000000000
  NORMAL 0.9775280899 0.7765957447 0.1746031746

>
> cat("\n")

>
```


Question 4.3 (c)
CPD table for 'HR' variable:

 Parameters of node HR (multinomial distribution)

Conditional probability table:

, , ERLO = FALSE

      HRBP
HR   HIGH   LOW   NORMAL
 HIGH   0.9950306701 0.0226395226 0.3444180523
 LOW   0.0004658747 0.0242190242 0.5415676960
 NORMAL 0.0045034552 0.9531414531 0.1140142518

, , ERLO = TRUE

      HRBP
HR   HIGH   LOW   NORMAL
 HIGH   0.0224719101 0.0744680851 0.8253968254
 LOW   0.0000000000 0.1489361702 0.0000000000
 NORMAL 0.9775280899 0.7765957447 0.1746031746

4.3) (D)

```
> # Q4.3 (d) Find the probability of P(HR = "HIGH" | BP = "LOW", PRSS = "NORMAL")
> cat("Question 4.3 (d)\n")
Question 4.3 (d)
>
> query_probability <- cpquery(bic_fitted_model, event = (HR == "HIGH"), evidence = (BP == "LOW" & PRSS == "NORMAL"))
> cat(sprintf("P(HR = 'HIGH' | BP = 'LOW', PRSS = 'NORMAL'): %f\n", query_probability))
P(HR = 'HIGH' | BP = 'LOW', PRSS = 'NORMAL'): 0.807448
>
```

```
Question 4.3 (d)
P(HR = 'HIGH' | BP = 'LOW', PRSS = 'NORMAL'): 0.807448
```

Q5)

(A)

The variables used in this analysis are:

1. **Sex** (male, female)
2. **Education** (unread, higher secondary, diploma, associate's degree, bachelor's degree, graduate degree, unknown)
3. **Vehicle type** (pickup, motorcycle, bus, minibus, truck, trailer, car, other)
4. **Age** (young, middle-aged, old, unknown)
5. **License type** (type 1, type2, expired, conditional, special, not seen, without license)
6. **Seatbelt status** (used, unused, unknown)
7. **Injury type** (not injured, injured, death)

The **PC (Parents and Children) algorithm** was employed for learning the Bayesian network structure from the road accident data.

(B)

In the learned Bayesian network provided in Figure 4, Injury Type is Conditionally independent of Sex given the knowledge about Education, Seat belt, License type and Vehicle type.

The Bayesian network shows that Sex influences Age, but since education, license type, vehicle type, and seatbelt are given an observed, All paths from sex to Injury type are blocked, therefore sex is conditionally independent from Injury type given education, license type, vehicle type, and seatbelt.

(C)

Figure 5 shows the unconditional (marginal) probabilities for each state of the variables. As stated in the paper, "By the unconditional term, we mean that the reported probabilities are those which describe each of the nodes regardless of the information of other nodes." For example, it shows 9.6% of drivers were female and 90.4% were male, irrespective of any other variables.

(D)

From the "Parameter learning in the road accident network" section:

In the notation used by the authors, π7 represents a specific parent configuration for the variable X7 (Injury Type). The set (X2, X3, X5, X6) makes up the parents of the node X7. Therefore, π7 = (i1, i2, i3, i4) denotes a particular combination of states for the parent variables Education (X2), Vehicle Type (X3), License Type (X5), and Seatbelt Status (X6).

I. The probability of being not injured while wearing a seat belt and driving a car, knowing that the driver has a diploma degree, and a type 2 driving license is 0.9822. This value was calculated as follows:

$$P(X7 = \text{not injured} \mid \pi7 = (3, 2, 6, 2)) = 1 - P(X7 = \text{injured} \mid \pi7 = (3, 2, 6, 2)) - P(X7 = \text{death} \mid \pi7 = (3, 2, 6, 2))$$
$$= 1 - 0.0166 - 0.0012 = 0.9822$$

Here, π7 = (3, 2, 6, 2) represents the parent configuration where Education (X2) = diploma (state 3), Vehicle Type (X3) = car (state 2), License Type (X5) = type 2 (state 6), and Seatbelt Status (X6) = used (state 2).

II. The probability of death while wearing a seat belt and driving a car, knowing that the driver has a diploma degree, and a type 2 driving license is 0.0012. This value was calculated as follows:

$$P(X7 = \text{death} \mid \pi7 = (3, 2, 6, 2)) = 0.0012$$

The authors walked through calculating these conditional probabilities using the maximum likelihood estimates based on the Bayesian network structure and the dataset. The key takeaway was that wearing a seatbelt significantly reduces the chances of injury and death compared to not wearing one.

Q6)

Introduction:

This report presents an analysis of traffic collisions in Montgomery County, USA, using Bayesian networks. The main objective is to investigate the relationships between various factors, including environmental variables, collision characteristics, and injury severity, by leveraging a combination of traffic collision data and weather data. Bayesian networks provide a probabilistic graphical model that allows for the identification and interpretation of dependencies among variables, making them suitable for this task [1].

Data Cleaning:

The analysis began with data cleaning and preprocessing steps. The traffic collision data was obtained from the "Crash_Reporting_-_Drivers_Data.csv" file, while the weather data was sourced from the "montgomery_data.csv" file. The relevant columns for the analysis were selected, including "Crash.Date.Time", "Collision.Type", "Weather", "Surface.Condition", "Light", "Injury.Severity", "Driver.Distracted.By", "Speed.Limit", and "Vehicle.Movement".

The following code snippet demonstrates the data cleaning process:

```
> # Filter crash data
> crash_data <- crash_data %>%
+   filter(Agency.Name == "Montgomery County Police") %>%
+   select(columns_for_csv)
> # Convert Crash.Date.Time to POSIXct format
> crash_data$Crash.Date.Time <- parse_date_time(crash_data$`Crash.Date.Time`, orders = "%m/%d/%Y %I:%M:%S %p", tz = "UTC")
> # Filter dates and convert to Date format
> crash_data <- crash_data %>%
+   filter(Crash.Date.Time >= as.Date("2023-01-01") & Crash.Date.Time <= as.Date("2023-12-31")) %>%
+   mutate(Crash.Date.Time = as.Date(Crash.Date.Time))
```

The "Crash.Date.Time" column was converted to the appropriate date format, and the data was filtered to include records from January 1, 2023, to December 31, 2023. This time period was chosen to focus on the most recent and complete year of data available. The weather data was preprocessed by converting imperial units to metric units and merging it with the crash data based on the "Crash.Date.Time" and "datetime" columns [2].

Table 1 presents the variables selected for the analysis along with their data types.

Table 1: Selected Variables and Data Types

Variable	Data Type
Collision.Type	Categorical
Surface.Condition	Categorical
Light	Categorical
Injury.Severity	Categorical
Driver.Distracted.By	Categorical
Speed.Limit	Categorical
Vehicle.Movement	Numeric
tempmax	Numeric
tempmin	Numeric
temp	Numeric
humidity	Numeric
windgust	Numeric
windspeed	Numeric
cloudcover	Numeric
visibility	Numeric
uvindex	Numeric

Categorical variables were converted to factors to ensure proper handling during the analysis. A subset of 16 variables was selected for the analysis, including both crash-related variables and weather variables such as temperature, humidity, wind speed, and visibility.

Exploratory Data Analysis:

Exploratory data analysis was performed to gain insights into the selected variables. Histograms were plotted for numeric variables, while bar plots were created for categorical variables (Image 1 and Image 2). Summary statistics were computed for both numeric and categorical variables to provide an overview of the data distribution and characteristics [3].

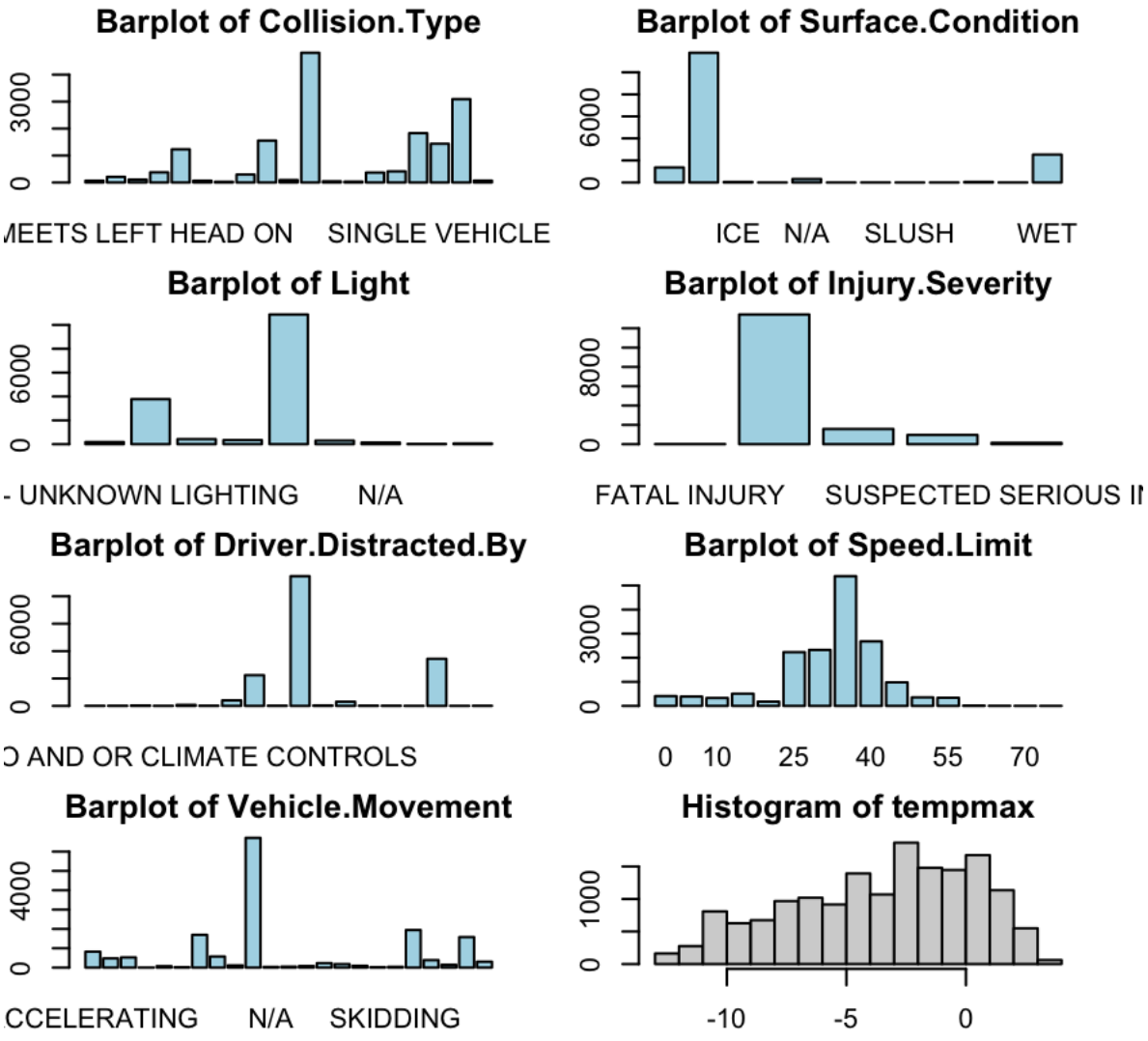
Key observations from the exploratory analysis include:

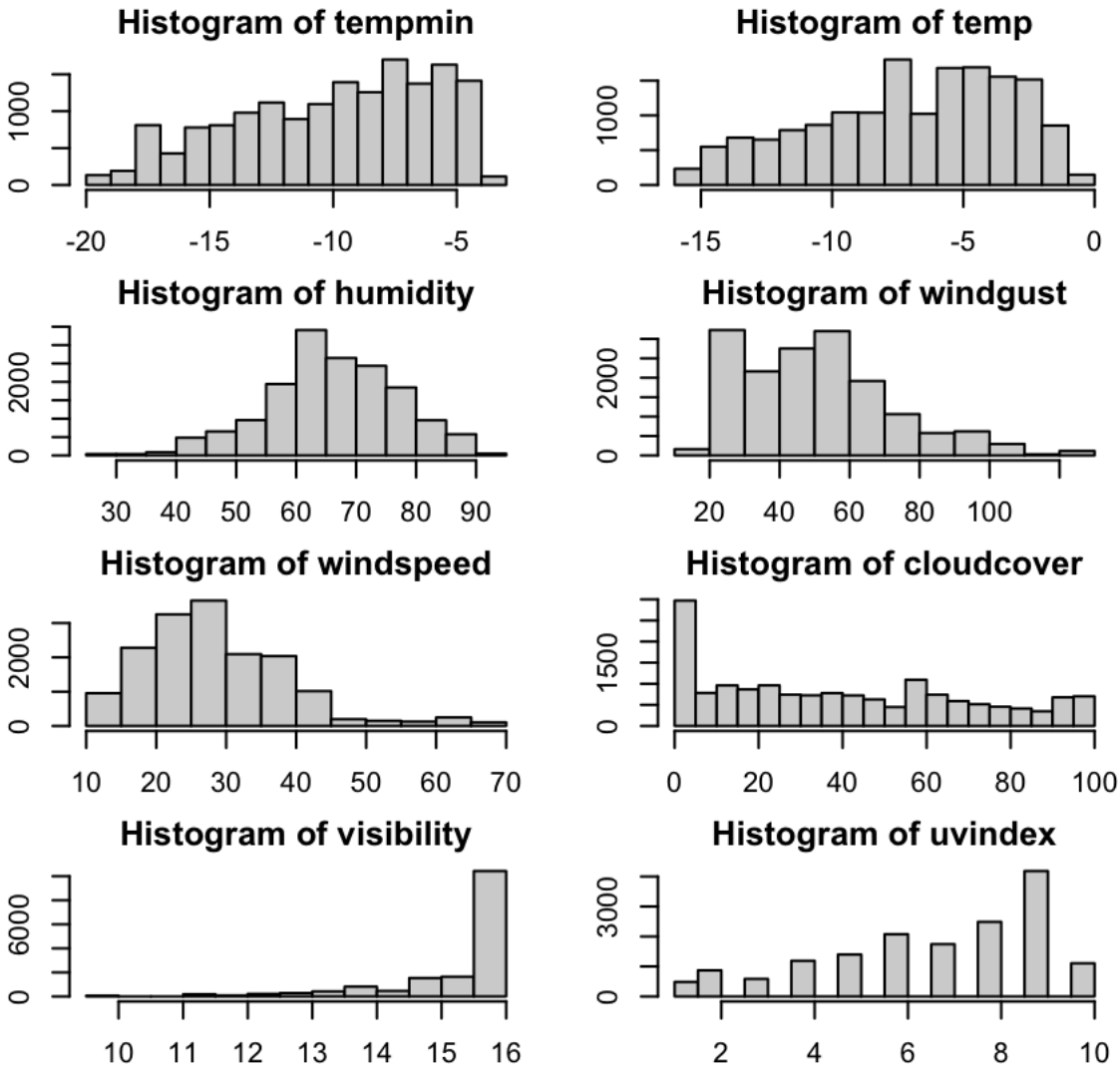
- The most common collision types were "SAME DIR REAR END" and "STRAIGHT MOVEMENT ANGLE".
- The majority of collisions occurred on dry surface conditions and during daylight.
- Most collisions resulted in no apparent injury or possible injury.

- The distribution of temperature variables (tempmax, tempmin, temp) was approximately normal.
- Humidity and wind-related variables (windgust, windspeed) showed right-skewed distributions.

The following code snippet demonstrates the exploratory analysis:

```
> # Perform exploratory analysis
> par(mfrow = c(4, 2), mar = c(2, 2, 2, 2))
> for (var in selected_variables) {
+   if (is.numeric(crash_data_subset[[var]])) {
+     hist(crash_data_subset[[var]], main = paste("Histogram of", var), xlab = var)
+   } else {
+     barplot(table(crash_data_subset[[var]]), main = paste("Barplot of", var), xlab = var, col = "lightblue")
+   }
+ }
>
```





Summary statistics were computed for both numeric and categorical variables to provide an overview of the data distribution and characteristics.

Bayesian Network Analysis

The main focus of the analysis was to learn the structure of Bayesian networks from the preprocessed data and perform parameter learning. The bnlearn package in R was used for this purpose [4]. Continuous variables were discretized using quantile-based discretization to enable their inclusion in the Bayesian network analysis [5].

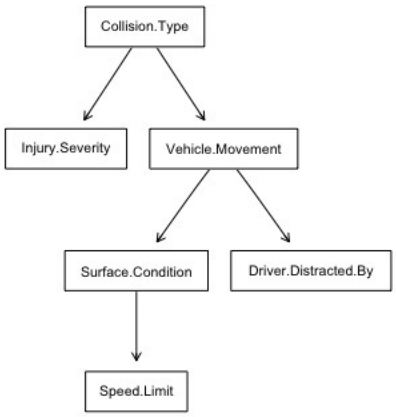
Several structure learning algorithms were employed to learn the Bayesian network structure from the data:

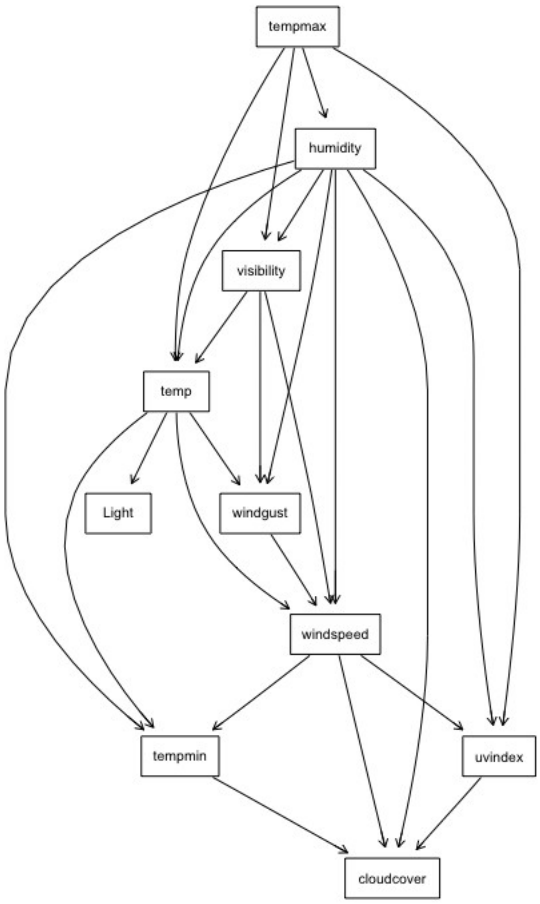
1. **Hill-Climbing (HC) algorithm:** A greedy search algorithm that starts with an empty network and iteratively adds, removes, or reverses edges to maximize a scoring function [6].

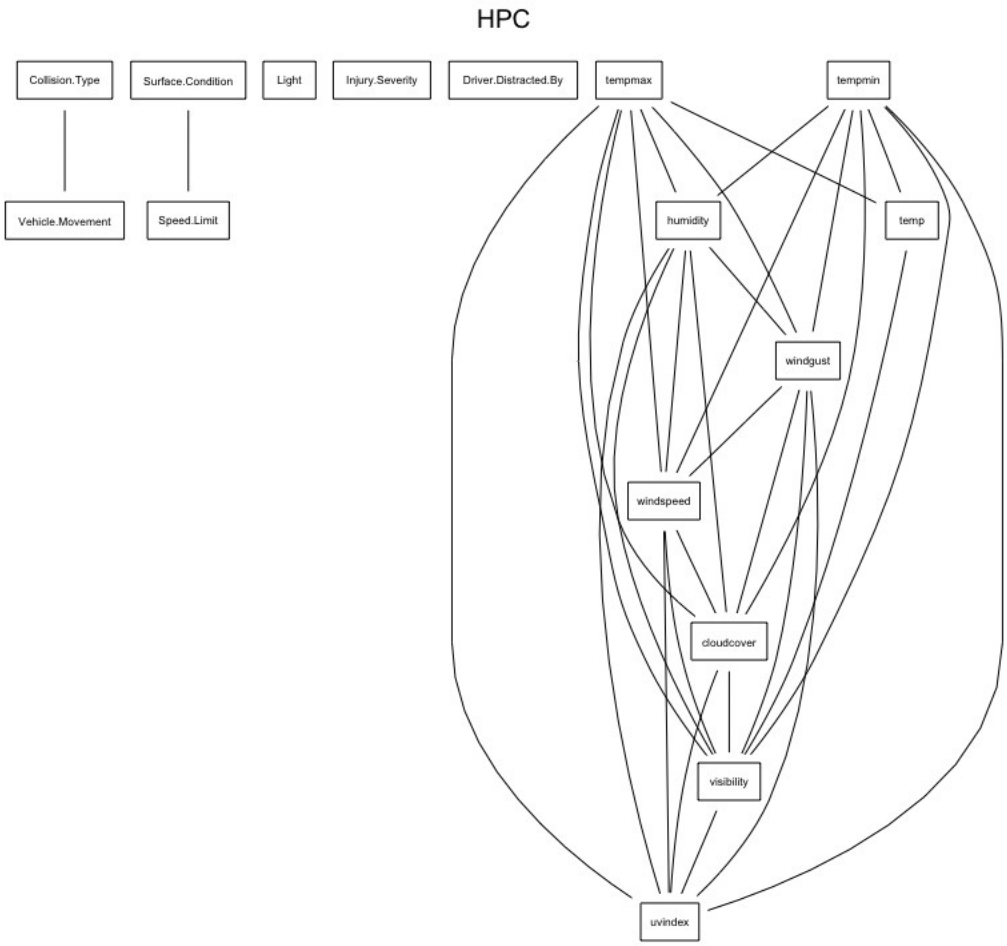
2. **Tabu search algorithm:** An extension of the Hill-Climbing algorithm that incorporates a tabu list to avoid revisiting recently explored network structures [7].
3. **Max-Min Hill-Climbing (MMHC) algorithm:** A hybrid algorithm that combines constraint-based and score-based approaches to learn the network structure [8].
4. **Hybrid Parents and Children (HPC) algorithm:** An algorithm that learns the network structure by iteratively identifying the parent and child nodes of each variable [9].

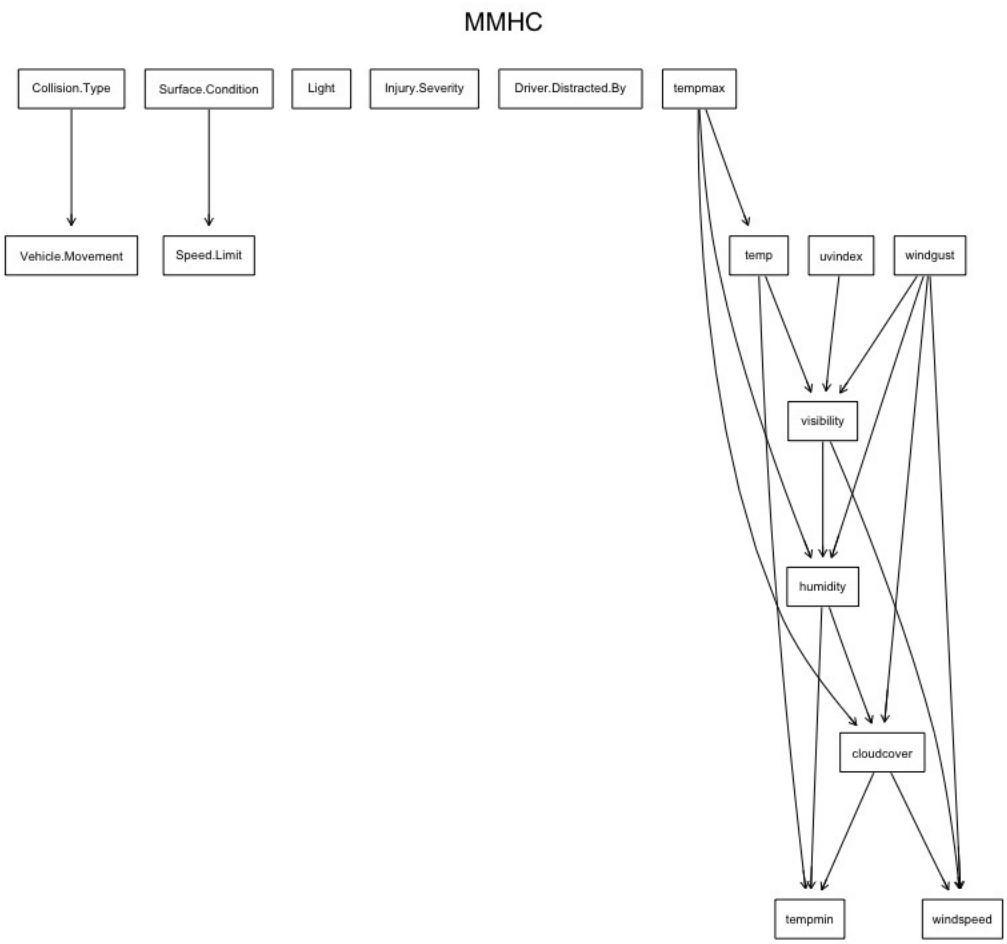
The learned network structures were visualized using the graphviz.plot() function to provide a graphical representation of the dependencies among variables (Image 3). Parameter learning was performed for each learned structure using the bn.fit() function to estimate the conditional probability distributions of the variables given their parents in the network [10].

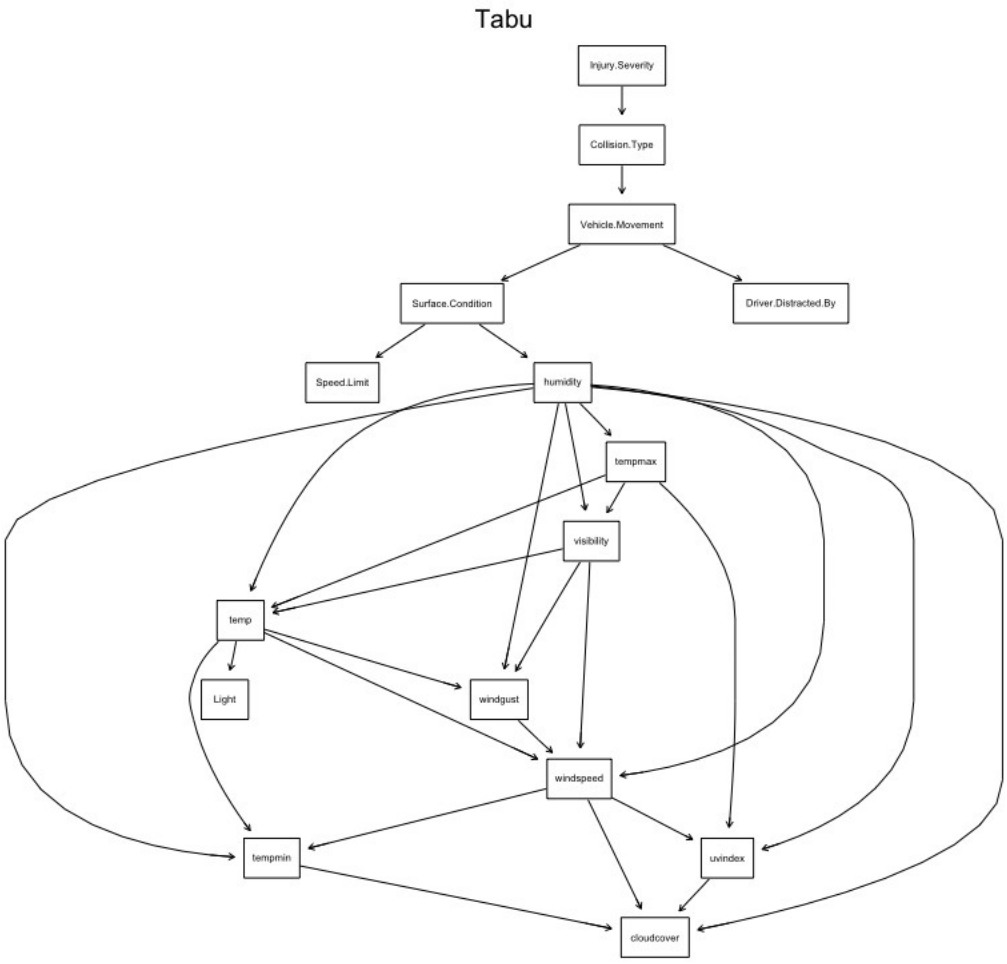
Hill-Climbing






HPC
Collision.Type
Surface.Condition
Light
Injury.Severity
Driver.Distracted.By
tempmax
tempmin
Vehicle.Movement
Speed.Limit
humidity
temp
windgust
windspeed
cloudcover
visibility
uvindex


MMHC
Collision.Type
Surface.Condition
Light
Injury.Severity
Driver.Distracted.By
tempmax
Vehicle.Movement
Speed.Limit
temp
uvindex
windgust
visibility
humidity
cloudcover
tempmin
windspeed



Interpretation of the learned network structures:

- In the Hill-Climbing network, "Collision.Type" appears to be a central node influencing several other variables, including "Injury.Severity" and "Vehicle.Movement".
- The Tabu search network shows similar relationships to the Hill-Climbing network, with additional edges connecting weather-related variables.
- The MMHC network exhibits a more complex structure, with multiple interconnected nodes and weather variables influencing collision characteristics.
- The HPC network presents a simpler structure compared to the other networks, focusing on the key relationships between collision-related variables.

Queries on the Bayesian Networks

Queries were conducted on the Bayesian networks to gain insights into the relationships between variables. The following queries were performed using the Hill-Climbing network:

1. Compute the probability: P(Injury Severity = "POSSIBLE INJURY" | Surface Condition = "WET")

```
> # Compute the probability: P(Injury Severity = "POSSIBLE INJURY" | Surface Condition = "WET")
> query1 <- cpquery(hc_bn_fitted, event = (Injury.Severity == "POSSIBLE INJURY"), evidence = (Surface.Condition == "WET"))
> print(query1)
[1] 0.101482
>
```

   The query result shows that the probability of injury severity being "POSSIBLE INJURY" given a wet surface condition is approximately 0.1014. This suggests that wet surface conditions may slightly increase the likelihood of possible injuries in collisions.

2. Check if "Injury Severity" is conditionally independent of "Speed Limit" given the learned network structure

```
> # Check if "Injury Severity" is conditionally independent of "Speed Limit" given the learned network structure
> query2 <- dsep(hc_bn, "Injury.Severity", "Speed.Limit")
> print(query2)
[1] FALSE
>
```

   The query result indicates that "Injury Severity" is not conditionally independent of "Speed Limit" in the learned Hill-Climbing network structure. This implies that there may be a relationship between speed limit and injury severity, either directly or through other intermediary variables.

3. Find the Markov blanket of "Driver Distracted By"

```
> # Find the Markov blanket of "Driver Distracted By"
> markov_blanket <- mb(hc_bn, "Driver.Distracted.By")
> print(markov_blanket)
[1] "Vehicle.Movement"
>
```

   The Markov blanket of the "Driver Distracted By" variable includes "Vehicle.Movement". This suggests that the variable "Vehicle.Movement" is directly related to driver distraction and plays a role in explaining its behaviour in the network.

Discussion on Variable Relationships

Based on the learned Bayesian network structures, several key relationships among the variables can be observed. The specific relationships and their strengths may vary depending on the algorithm used and the characteristics of the dataset.

In general, the Bayesian networks reveal dependencies and probabilistic relationships among the collision-related variables and weather variables. For example, the networks show connections between collision type, surface condition, light condition, and injury severity, indicating that these factors may influence each other. Weather variables such as temperature, humidity, wind speed, and visibility also appear to have an impact on collision characteristics and outcomes [1].

It is important to interpret the learned relationships in the context of domain knowledge and provide insights into how these factors may contribute to traffic collisions and injury severity. The discussion should also consider any notable differences or similarities in the relationships learned by different algorithms [4].

Conclusion

The analysis demonstrates the application of Bayesian networks to investigate the relationships between various factors related to traffic collisions in Montgomery County. By learning the structure of Bayesian networks from the data and performing parameter learning, insights can be gained into the dependencies and probabilistic relationships among the variables.

The report highlights the data cleaning and preprocessing steps, exploratory data analysis, and the implementation of multiple structure learning algorithms. The learned network structures provide a visual representation of the dependencies, while queries on the networks allow for further exploration of specific relationships.

Overall, this analysis serves as a foundation for understanding the complex interplay of factors influencing traffic collisions and injury severity. The insights gained from the Bayesian network analysis can potentially inform decision-making and policy development related to road safety and accident prevention in Montgomery County [6].

References

[1] Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann.

[2] Wickham, H. (2014). Tidy data. Journal of Statistical Software, 59(10), 1-23.

[3] Tukey, J. W. (1977). Exploratory Data Analysis. Addison-Wesley.

[4] Scutari, M. (2010). Learning Bayesian networks with the bnlearn R package. Journal of Statistical Software, 35(3), 1-22.
[5] Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In Machine Learning Proceedings 1995 (pp. 194-202). Morgan Kaufmann.
[6] Tsamardinos, I., Brown, L. E., & Aliferis, C. F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, 65(1), 31-78.
[7] Glover, F. (1989). Tabu search—part I. ORSA Journal on Computing, 1(3), 190-206.
[8] Tsamardinos, I., Brown, L. E., & Aliferis, C. F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, 65(1), 31-78.
[9] Gámez, J. A., Mateo, J. L., & Puerta, J. M. (2011). Learning Bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood. Data Mining and Knowledge Discovery, 22(1-2), 106-148.
[10] Koller, D., & Friedman, N. (2009). Probabilistic Graphical Models: Principles and Techniques. MIT Press.