Student ID: 201503267

1. **Learn different Bayesian Networks and evaluate their capability of predicting, which persons are more likely to earn over $50K/year. Evaluate the networks predictive power using different performance measures. (Note: ROC/AUC from k-fold cross-validation is not required).**

Answer: We know that Special purposes networks (Naïve and TAN) are used for classification the target variable. Here the target variable is if the person earns more than $50K/year or not. So we can use these structures for our data, and finally test how good they can predict the target variable (regarding the arcs – relations between target variable and all other variables).

After that, we can use other algorithms for leaning the structure and the joint probabilities of the variables using factorization algorithms (constrained-based and score-based), so also we use them.

Furthermore, we can use resampling methods to repeat the way learning algorithms use, and then using threshold of directions and also strength of arcs to discover the averaged DAG (directed acyclic graph).

At the end we use evaluation algorithms for discovering which structure is more powerful to predict the target variable.

We do all of it in R Studio.

call the bnlearn packageName, also other packages we need to evaluate different structures
library (bnlearn)
library (gRain)
library (gRbase)
library (caTools)

after that we use tools -> import dataset of R Studio for importing the datasets.

# Copy the data set in another one to keep the original
Data1 <- Data
# Getting a sense about the data - if there is null values?
summary(Data1)

```
           V1                      V2                            V3                  
V4                
 (41.3,65.7]:16153   Federal-gov     : 1406   (5.06e+05,9.98e+05]:  458   HS-grad     
:14783  
 (65.7,90.1]: 1344   Local-gov       : 3100   (9.98e+05,1.49e+06]:   17   Some-colle
ge: 9899  
 [16.9,41.3]:27725   Private         :33307   [1.2e+04,5.06e+05] :44747   Bachelors   
: 7570  
                     Self-emp-inc    : 1646                               Masters     
: 2514  
                     Self-emp-not-inc: 3796                               Assoc-voc   
: 1959  
                     State-gov       : 1946                               11th        
: 1619  
                     Without-pay     :   21                               (Other)     
: 6878  
         V5                          V6                       V7                  
V8        
 (11,16]  :12920   Divorced             : 6297   Craft-repair   : 6020   Amer-Indian
-Eskimo:  435  
 (6,11]   :28837   Married-AF-spouse    :   32   Prof-specialty : 6008   Asian-Pac-I
slander: 1303  
 [0.985,6]: 3465   Married-civ-spouse   :21055   Exec-managerial: 5984   Black       
: 4228  
                   Married-spouse-absent:  552   Adm-clerical   : 5540   Other       
:  353  
```

```
                   Never-married        :14598   Sales          : 5408   White
:38903
                   Separated            : 1411   Other-service  : 4808
                   Widowed              : 1277   (Other)        :11454
      V9                          V10                        V11                     V12
 Female:14695   (3.33e+04,6.67e+04]:    7   (1.45e+03,2.9e+03]: 2002   (33.7,66.3] :
37089
 Male  :30527   (6.67e+04,1e+05]   :  229   (2.9e+03,4.36e+03]:   15   (66.3,99.1] :
1156
                [-100,3.33e+04]    :44986   [-4.36,1.45e+03]  :43205   [0.902,33.7]:
6977
```

```
           V13            V14
 United-States:41292   <=50K:34014
 Mexico       :  903   >50K :11208
 Philippines  :  283
 Germany      :  193
 Puerto-Rico  :  175
 Canada       :  163
 (Other)      : 2213
```

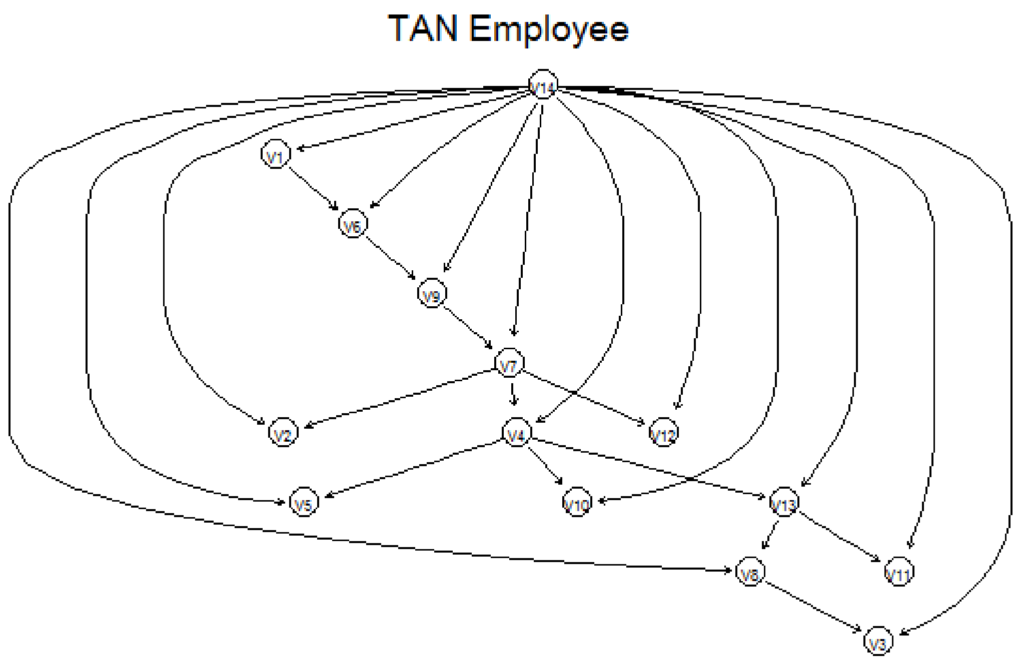
As we see, all data do not have null values and also all variables are categorical (so there is no need to discretze them)

So we do not have any null value (N/A) in our dataset

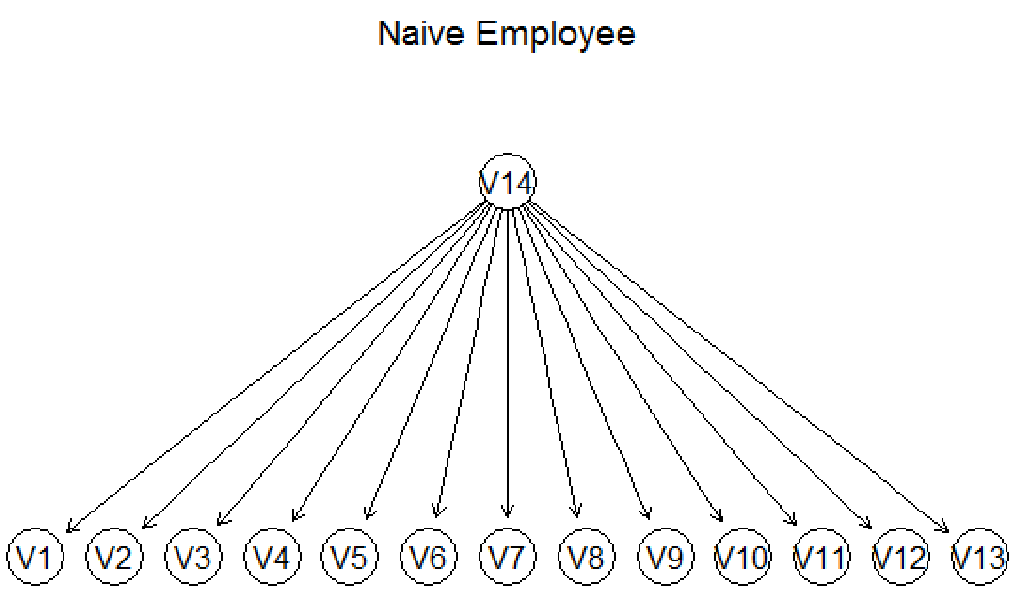
# if they are factors or numbers?
str(Data1)

They are all factors, so they have been discretized before

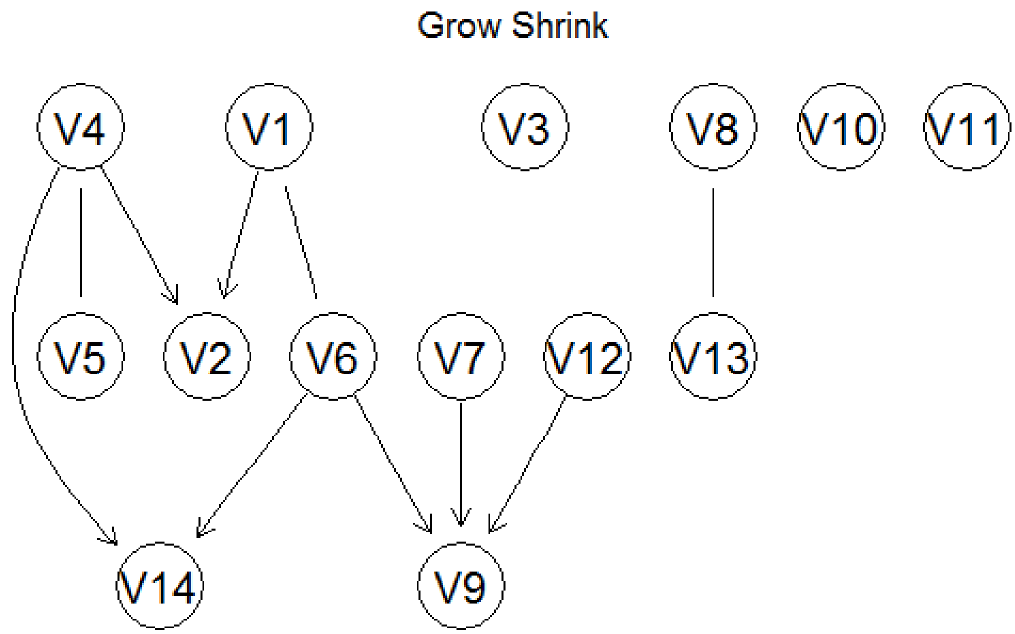
# We select the v14 (boolean value) as our target value in TAN structure
# 1 - Learning TAN structure
emp.tan <- tree.bayes(Data1, "V14")
graphviz.plot(emp.tan, main = "TAN Employee")



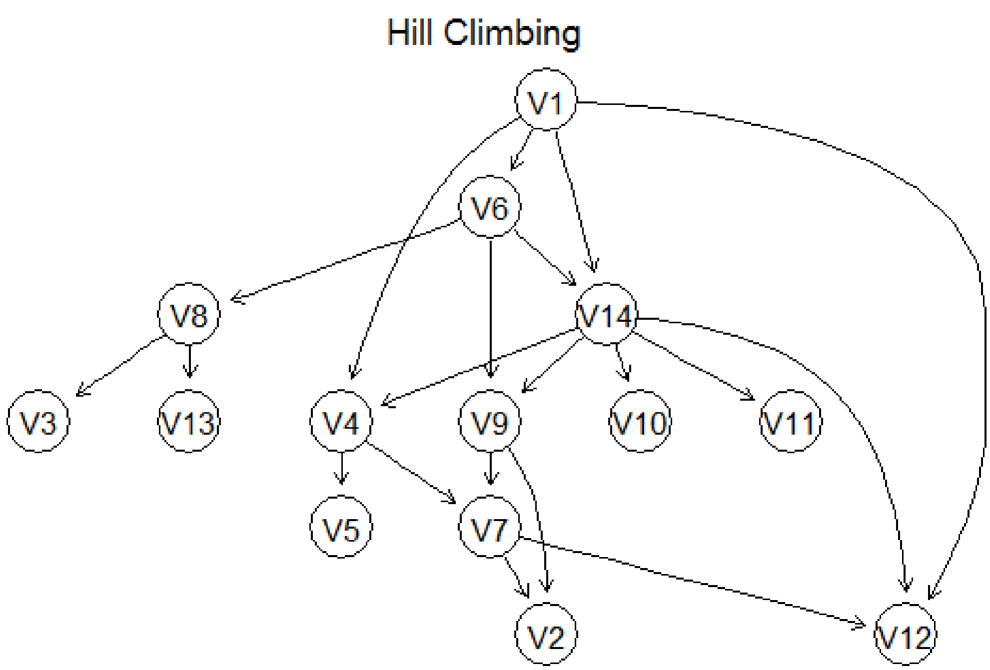

```
# Also we can use naive bayesian network to learn the structure
# 2 - Learning Naive Bayesian structure
emp.nb <- naive.bayes(Data1, "V14")
graphviz.plot(emp.nb, main = "Naive Employee")
```




```
# In TAN structure the nodes can be related to each other

# 3 - Constrained-based algorithms (Grow-Shrink)
emp.gs <- gs(Data1, alpha = 0.05, test = "x2") # alpha = the sig. level
graphviz.plot(emp.gs, main = "Grow Shrink")
```

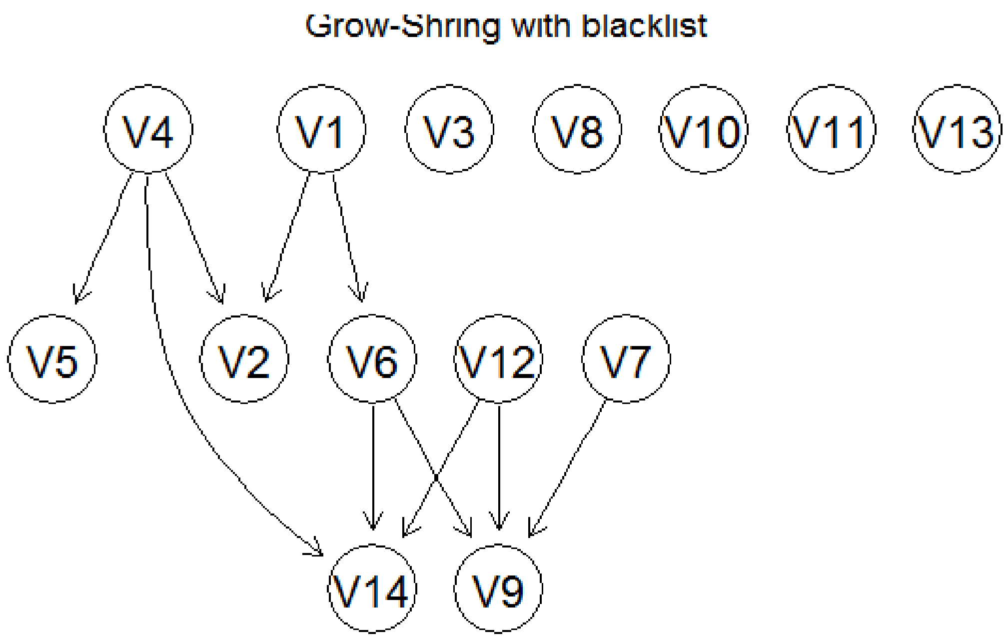



```
# 4 - Score-based algorithms (Hill-Climbing)
emp.hc <- hc(Data1, score = "bic")
graphviz.plot(emp.hc, main = "Hill Climbing")
```

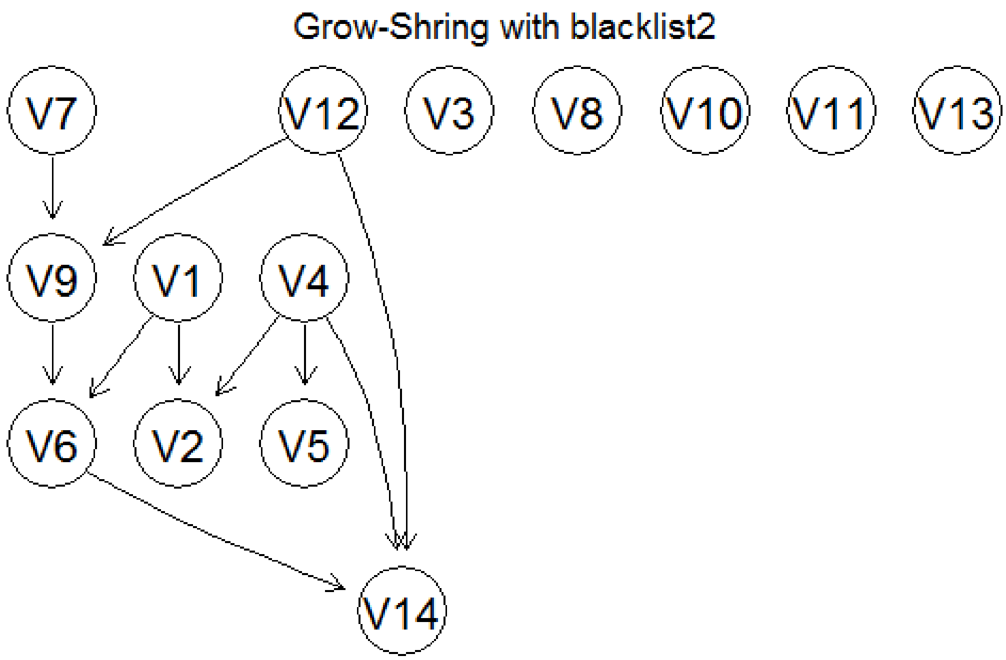



```
# In the constrained-based graphs some links are undirected (the software cannot
# establish the direction of the causality).
undirected.arcs(emp.gs)


# We use Hill-Climbing structure to put undirected arcs into black list
# And also use Grow-Shrink plot to put other undirected arcs into black list
blacklist = data.frame(from = c("V6","V5","V13","V14","V14","V14","V9"),
```

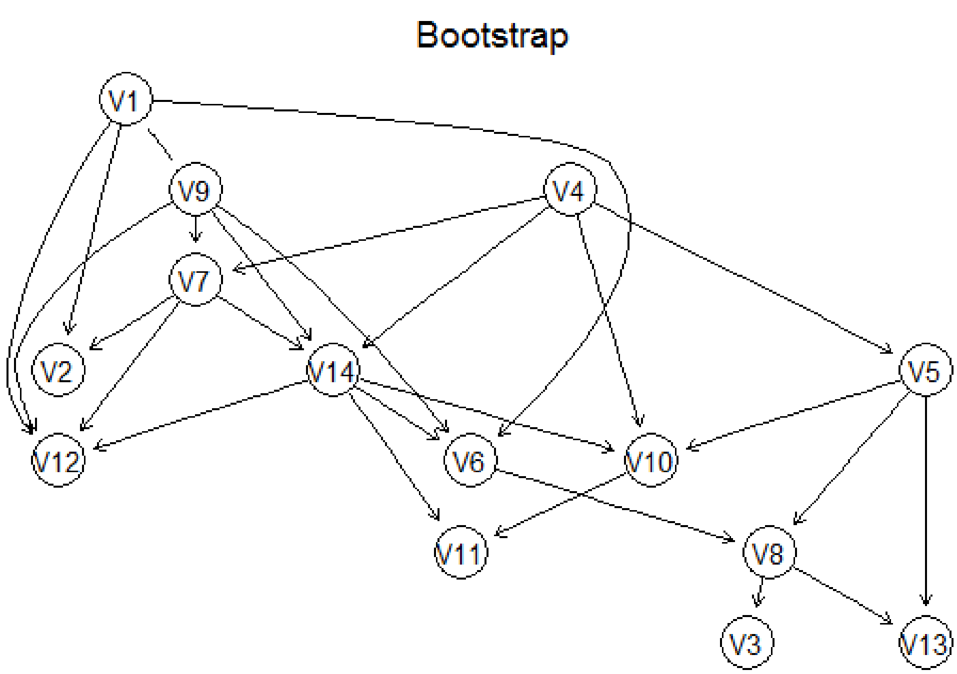
```
                to   = c("V1","V4","V8" ,"V6" ,"V4" ,"V12","V6"))
# Some of these arcs are rational, so we can decide easily about the deleting arcs in opposite directions:
# Rational arcs: V1 -> V6 : Age -> Marital-status
#             V4 -> V5 : edu level -> edu years
#             V13 -> V8: country -> Race
# about V14, of course we assume other factors are affecting the revenue
#             V9 -> V6 : Sex -> Marital status
# So our other option for Grow-Shrink DAG is:
blacklist2 = data.frame(from = c("V6","V5","V8","V14","V14","V14","V6"),
                to   = c("V1","V4","V13","V6","V4" ,"V12","V9"))
blacklist
blacklist2
emp.gsb <- gs(Data1, blacklist = blacklist)
graphviz.plot(emp.gsb, main = "Grow-Shring with blacklist")
```




```
emp.gsb2 <- gs(Data1, blacklist = blacklist2)
graphviz.plot(emp.gsb2, main = "Grow-Shring with blacklist2")
```




```
# 5 - Bootstrap (using boot.strength function) - learning from data by averaging multiple DAGs
# number of samples: 500
# algorithm: Hill-climbing for learning each sample
# method: bde (we assume 10 instance of our experience, with the same probability for each state)
boot <- boot.strength(Data1, R = 500, algorithm = "hc",
                algorithm.args = list(score = "bde", iss = 10))
# The arcs which are stronger than 0.85 and the probability of their direction > 0.5
boot [(boot$strength > 0.85) & (boot$direction >= 0.5), ]
# The averaged network
emp.boot <- averaged.network (boot, threshold = 0.85)
graphviz.plot(emp.boot, main = "Bootstrap")
```



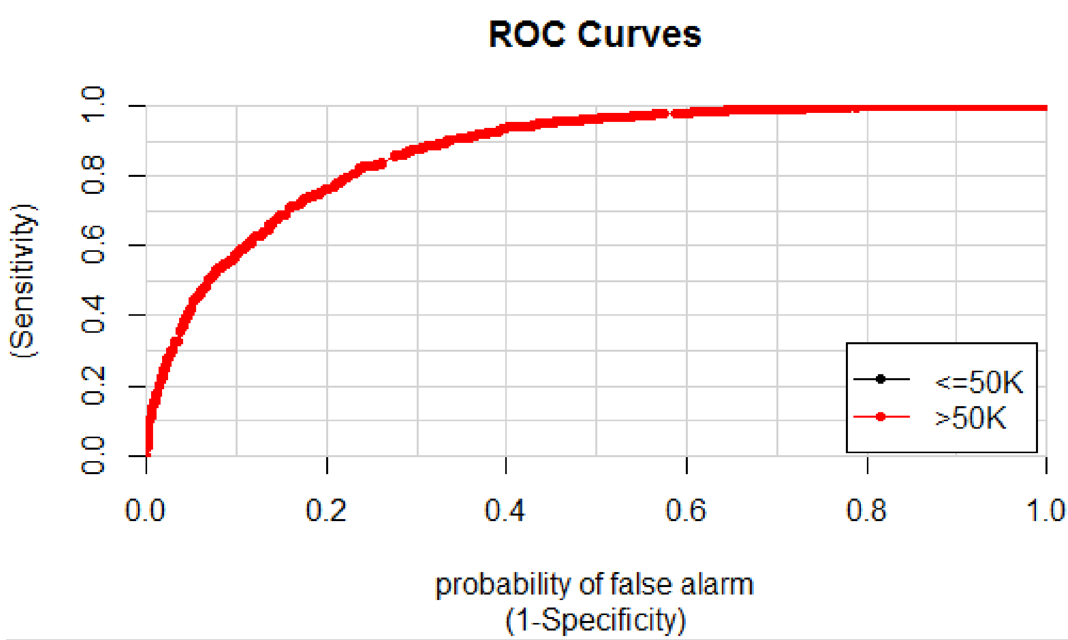
For evaluating the DAGs in terms of their performance in prediction the V14 variable, we use k-fold cross-validation algorithm.

```
# Evaluating the networks (Misclassification Error) using k-fold cross-validation
# We are going to perform a 5-fold cross validation for all structures
# using the classification error for V14 as a loss function.
nbcv = bn.cv(Data1, emp.nb, loss = "pred", k = 5, loss.arg = list(target = "V14"))
nbcv   # 0.19
tancv = bn.cv(Data1, emp.tan, loss = "pred", k = 5, loss.arg = list(target = "V14"))
tancv  # 0.17
gscv = bn.cv(Data1, emp.gsb, loss = "pred", k = 5, loss.arg = list(target = "V14"))
gscv   # 0.1829862
gscv2 = bn.cv(Data1, emp.gsb2, loss = "pred", k = 5, loss.arg = list(target = "V14"))
gscv2   # 0.1827208
hccv = bn.cv(Data1, emp.hc, loss = "pred", k = 5, loss.arg = list(target = "V14"))
hccv   # 0.24
bootcv = bn.cv(Data1, emp.boot, loss = "pred", k = 5, loss.arg = list(target = "V14"))
bootcv # 0.21

# TAN structure has the best prediction performance in terms of predicting the
# target variable and misclassification rate. After that the Grow-Shrink
# structures (Second is a little better) and Naive are the best
# So we can conclude that classification algorithms perform well in prediction of
# target and having low values in misclassification rates

# Evaluating the networks (Prediction Performance) using AUC and ROC curve
# We use predict function for the best cross-validation structure as well as
# all structures we have (Naive, TAN, GS, HC, Bootstrap)

# TAN
netcvfit1 = as.grain(tancv[[1]]$fitted)
emp_test1 = Data1[tancv[[1]]$test, ]
pred_test1 = predict(netcvfit1, response = c("V14"), newdata = emp_test1,
                predictors = names(emp_test1)[-14], type = "distribution")

# Here we encountered to error smooth argument and exit
# We continue with other networks:

# Copy the data set in another one to keep the original
Data2 <- Datatest

# Getting a sense about the data - if there is null values?
summary(Data2)

# So we do not have any null value (N/A) in our dataset

# if they are factors or numbers?
str(Data2)
```

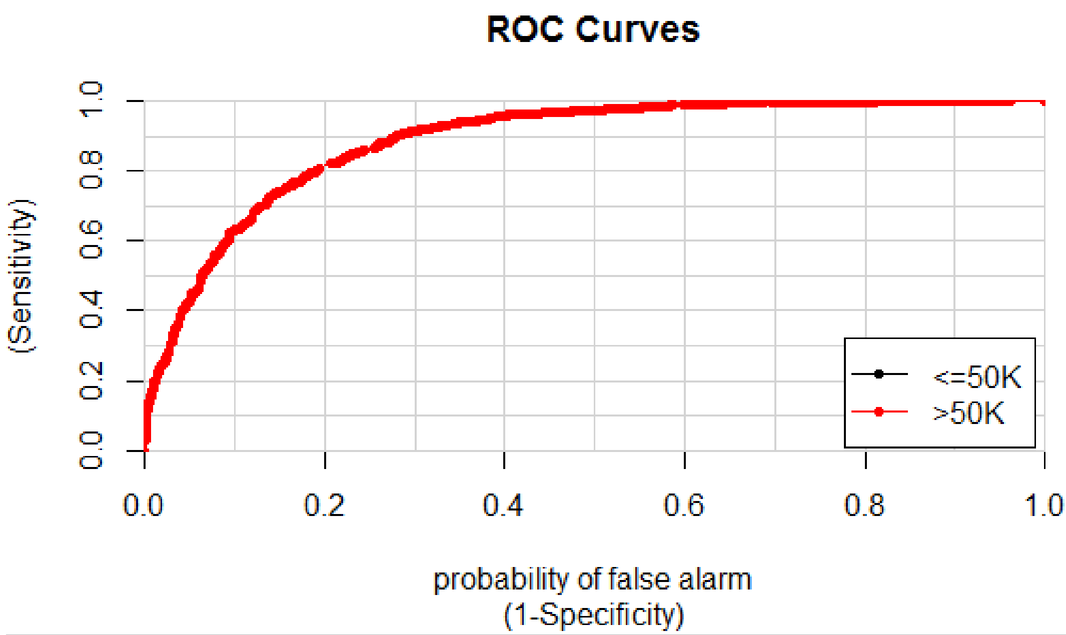
```
'data.frame':   3504 obs. of  14 variables:
 $ V1 : Factor w/ 3 levels "(41.3,65.7]",..: 3 1 3 3 1 1 3 3 1 1 ...
 $ V2 : Factor w/ 7 levels "Federal-gov",..: 3 3 3 3 3 3 1 3 3 3 ...
 $ V3 : Factor w/ 2 levels "(5.06e+05,9.98e+05]",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ V4 : Factor w/ 16 levels "10th","11th",..: 2 16 1 16 6 12 10 12 12 13 ...
```

```
 $ V5 : Factor w/ 3 levels "(11,16]","(6,11]",..: 2 2 3 2 3 2 1 2 2 1 ...
 $ V6 : Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 5 3 5 5 3 3 3 5 3 3 ..
.
 $ V7 : Factor w/ 14 levels "Adm-clerical",..: 7 7 8 8 3 7 1 1 7 4 ...
 $ V8 : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 3 3 5 5 5 5 5 5 5 5 ...
 $ V9 : Factor w/ 2 levels "Female","Male": 2 2 2 1 2 2 2 1 2 2 ...
 $ V10: Factor w/ 3 levels "(3.33e+04,6.67e+04]",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ V11: Factor w/ 3 levels "(1.45e+03,2.9e+03]",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ V12: Factor w/ 3 levels "(33.7,66.3]",..: 1 1 3 1 3 1 1 1 1 1 ...
 $ V13: Factor w/ 38 levels "Cambodia","Canada",..: 36 36 36 36 36 36 36 36 36 36 ..
.
 $ V14: Factor w/ 2 levels "<=50K",">50K": 1 2 1 1 1 2 1 1 2 2 ...
```

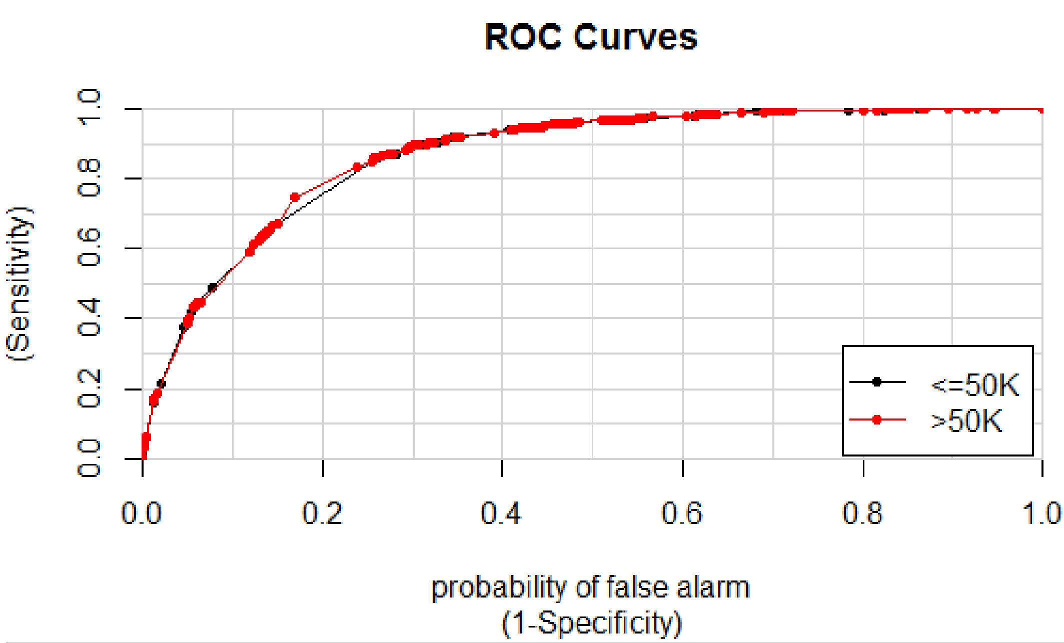
```
# Evaluating the networks (Prediction Performance) using AUC and ROC curve (Cont.)
# This time we use the Data2 as a test dataset

# AUC and ROC curve for Naive
fitted_nb = bn.fit(emp.nb, Data1)
nb_fit <- as.grain(fitted_nb)
pred_nb = predict(nb_fit, response = c("V14"), newdata = Data2,
            predictors = names(Data2)[-14], type = "distribution")
colAUC(pred_nb$pred$V14, Data2[, 14], plotROC = TRUE)
#              <=50K     >50K
# <=50K vs. >50K 0.8710928 0.8710928
```




```
# AUC and ROC curve for TAN
fitted_tan = bn.fit(emp.tan, Data1)
tan_fit <- as.grain(fitted_tan)
pred_tan = predict(tan_fit, response = c("V14"), newdata = Data2,
           predictors = names(Data2)[-14], type = "distribution")
colAUC(pred_tan$pred$V14, Data2[, 14], plotROC = TRUE)
#              <=50K     >50K
# <=50K vs. >50K 0.8880065 0.8880065
```

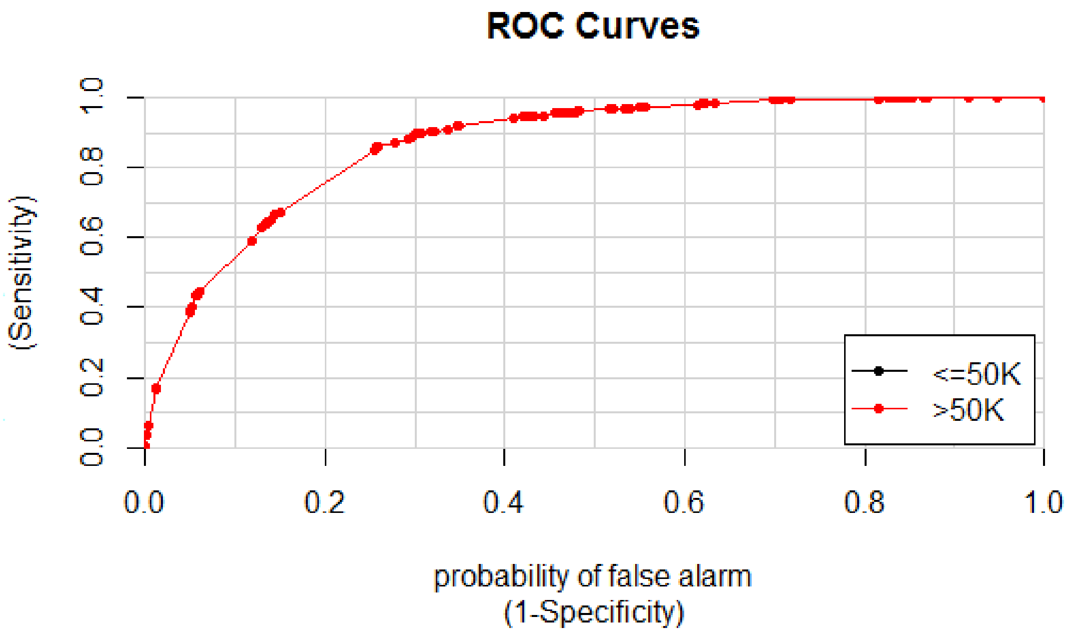



```
# AUC and ROC curve for GS
fitted_gs = bn.fit(emp.gsb, Data1)
gs_fit <- as.grain(fitted_gs)
pred_gs = predict(gs_fit, response = c("V14"), newdata = Data2,
              predictors = names(Data2)[-14], type = "distribution")
colAUC(pred_gs$pred$V14, Data2[, 14], plotROC = TRUE)
#                <=50K      >50K
# <=50K vs. >50K 0.8681308 0.8697158
```

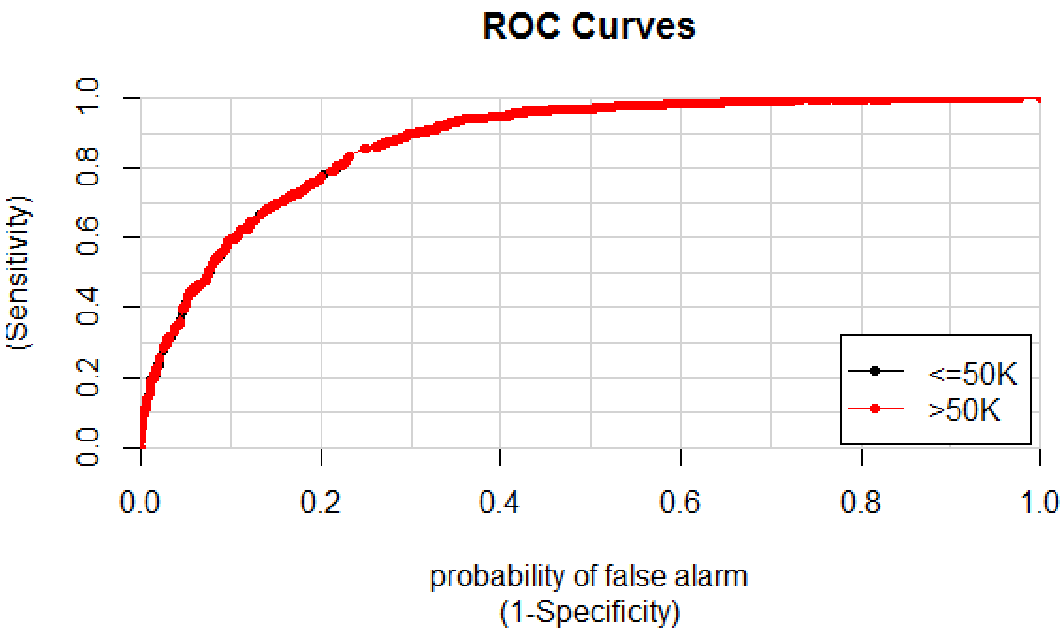



```
# AUC and ROC curve for GS
fitted_gs2 = bn.fit(emp.gsb2, Data1)
```

```
gs_fit2 <- as.grain(fitted_gs2)
pred_gs2 = predict(gs_fit2, response = c("V14"), newdata = Data2,
            predictors = names(Data2)[-14], type = "distribution")
colAUC(pred_gs2$pred$V14, Data2[, 14], plotROC = TRUE)
#              <=50K     >50K
# <=50K vs. >50K 0.8677789 0.8677789
```




```
# AUC and ROC curve for HC
fitted_hc = bn.fit(emp.hc, Data1)
hc_fit <- as.grain(fitted_hc)
pred_hc = predict(hc_fit, response = c("V14"), newdata = Data2,
            predictors = names(Data2)[-14], type = "distribution")
colAUC(pred_hc$pred$V14, Data2[, 14], plotROC = TRUE)
#              <=50K     >50K
# <=50K vs. >50K 0.8769131 0.8769433
```



Note: We cannot use "grain" function to transform a bn which is built based on
cpt to a junction tree

TAN structure has the best prediction performance in terms of AUC and ROC curve
(Just like Cross-validation). In fact the percentage of correct prediction
in TAN (for both categories: <= 50K and > 50K) is more than other BNs. After
that the Hill-Climbing and Naive are the best with Data2 dataset as our test set.

2. **Describe, which are the variables that best explain whether or not a person makes over 50K a year. The descriptions and accompanying interpretation must be comprehensible for somebody with no prior knowledge of BNs.**

We can use two measure for this purpose. Sensitivity Analysis and Influence Analysis.

1. Sensitivity Analysis – using arc.strength – each variable with the most strong arc means more dependency between target variable and it:

# comparison between arcs (Naive)
arc.strength(emp.nb, data = Data1, criterion = "x2")

```
X1  X2      strength
1  V14  V1  0.000000e+00
2  V14  V2 1.284668e-257
3  V14  V3  7.022316e-01
4  V14  V4  0.000000e+00
5  V14  V5  0.000000e+00
6  V14  V6  0.000000e+00
7  V14  V7  0.000000e+00
8  V14  V8  1.379446e-96
9  V14  V9  0.000000e+00
10 V14 V10 6.852522e-153
11 V14 V11 2.711663e-216
12 V14 V12 5.325089e-319
13 V14 V13  5.399311e-72

The three first (powerful) arcs are: V12 (working hours per week)
```

```
V2: Work class
V11: Capital loss

# comparison between arcs (TAN)
arc.strength(emp.tan, data = Data1, criterion = "x2")

   from  to      strength
1   V14  V1  0.000000e+00
2   V14  V2 8.433049e-145
3   V14  V3  5.184579e-01
4   V14  V4  0.000000e+00
5   V14  V5  1.000000e+00
6   V14  V6  0.000000e+00
7   V14  V7  0.000000e+00
8   V14  V8  4.916812e-36
9   V14  V9  5.149030e-54
10  V14 V10  6.574976e-72
11  V14 V11 9.075540e-162
12  V14 V12 6.291044e-206
13  V14 V13  5.880908e-01
14   V1  V6  0.000000e+00
15   V7  V2  0.000000e+00
16   V8  V3  1.739765e-47
17   V4  V5  0.000000e+00
18   V7  V4  0.000000e+00
19   V4 V10  7.564440e-52
20   V4 V13  0.000000e+00
21   V6  V9  0.000000e+00
22   V9  V7  0.000000e+00
23   V7 V12  0.000000e+00
24  V13  V8  0.000000e+00
25  V13 V11  2.498457e-02

In order: V12, V11, V2 (the same nodes)
```

arc.strength(emp.gsb, data = Data1, criterion = "x2") # all arcs are strong enough - useful to discover casualities

```
   from  to      strength
1    V1  V2 1.046954e-308
2    V1  V6  0.000000e+00
3    V4  V2  0.000000e+00
4    V4  V5  0.000000e+00
5    V4 V14  0.000000e+00
6    V6  V9  0.000000e+00
7    V6 V14  0.000000e+00
8    V7  V9  0.000000e+00
9   V12  V9  1.550258e-93
10  V12 V14  1.463513e-67
```

In order: V6 (Marital status), V4 (education), V12

arc.strength(emp.hc, data = Data1, criterion = "x2") # all arcs are strong enough - useful for prediction

```
   from  to      strength
1    V4  V5  0.000000e+00
2    V4  V7  0.000000e+00
3    V6  V9  0.000000e+00
4    V1  V6  0.000000e+00
5    V6 V14  0.000000e+00
6    V7  V2  0.000000e+00
7    V9  V7  0.000000e+00
8    V8 V13  0.000000e+00
9   V14  V4  0.000000e+00
10   V7 V12  0.000000e+00
```

```
11   V1 V12  0.000000e+00
12   V6  V8 8.950331e-241
13  V14 V11 2.711663e-216
14   V1 V14 5.780720e-187
15   V1  V4 7.019292e-263
16  V14 V10 6.852522e-153
17  V14 V12 3.683500e-156
18  V14  V9  5.149030e-54
19   V9  V2 9.089334e-161
20   V8  V3  4.662356e-52

No new nodes.
So the most affective factors to have a revenue of $50K/year or not are these factor
s:
V12 (working hours per week)
V2: Work class
V11: Capital loss
```

V6 (Marital status),
V4 (education),

```
# Influence Analysis: Now we can test the nodes we have found from the
# Sensitivity analysis: we use TAN structure because it had the best performance in
prediction the target variable
fitted <- bn.fit(emp.tan, Data1, method = "mle")
fitted

fitted$V12
fitted$V2
fitted$V14

str(Data1)

# We use cpquery function to get the approximate inference and see how the target va
riable has sensitivity to the selected variables

# Because we do not have enough time, we are going to select a sample and caannot se
e all joint and conditional probabilities for V14
# Also we have selected V12 for our sample

> cpquery(fitted, event = (V14 == "<=50K"), evidence = (V12 == "(33.7,66.3]"))
[1] 0.7232303
> cpquery(fitted, event = (V14 == "<=50K"), evidence = (V12 == "(66.3,99.1]"))
[1] 0.6991643
> cpquery(fitted, event = (V14 == ">50K"), evidence = (V12 == "(33.7,66.3]"))
[1] 0.2821597
> cpquery(fitted, event = (V14 == ">50K"), evidence = (V12 == "(66.3,99.1]"))
[1] 0.3711048
```

As we can see changes in states of V12 (working hours), has more effect on revenues more than $50K (9%) but it has less effect on revenues less than $50K (3%)

We can test other variables and states and see which of them has the most effect on V14
Also we can use combination of variables to see the effects of changing them on V14

3. Give an example of prediction and diagnostic inference using your network.

################### Question 3 #######################

We have seen the approximate inference in question 2 (cpquery)
Now we can test Exact inference with querygrain command (function)

```
# Transform the bn into a junction tree (using "as.grain" function) and compute
# probability tables ("compile" function)
# In this example we use Hill-Climbing structure
bn <- bn.fit(emp.hc, data = Data1)
# Transform the bn into a junction tree (using "as.grain" function) and compute
# probability tables ("compile" function)
junction <- compile(as.grain(bn))
junction

querygrain(junction, nodes = "V14")$V14 # Exact Inference
# This shows that about 75% of employees (our sample) have the revenue of $50K per year and 25% of them have more than $50K

# Let see how changing the states of other variables can affect this percentages
# for example sex
str(Data1)
jsex <- setEvidence(junction, nodes = "V9", states = "Female")
querygrain(jsex, nodes = "V14")$V14 # Show the probability
V14
   <=50K     >50K
0.886424 0.113576

# We can see if employee is a woman, the percentage of getting more than $50K per year can decreased to just 11%
# In other words just 11% of all woman of our sample get more than $50K each year
# Let try with men
jsex <- setEvidence(junction, nodes = "V9", states = "Male")
querygrain(jsex, nodes = "V14")$V14 # Show the probability
# We see we have an increase of 6% (from 25% to 31%) for receiving more than $50K if the sample is a man
V14
    <=50K      >50K
0.6875225 0.3124775

# The default value displayed in the querygrain is marginal probability
# distribution. but we can also display the joint or conditional.
# e.g. P(V14, V9 | V13 = Canada)
jcon <- setEvidence(junction, nodes = "V13", states = "Canada")
SxT.cpt <- querygrain(jcon, nodes = c("V14", "V9"), type = "joint")
SxT.cpt
        V14
V9           <=50K       >50K
  Female 0.2790709 0.03747479
  Male   0.4650357 0.21841865
# This shows in Canada how the percentages are distributed

# Also we can use conditional probabilities
SxT.cpt <- querygrain(jcon, nodes = c("V14", "V9"), type = "conditional")
SxT.cpt
        V14
V9           <=50K      >50K
  Female 0.8816133 0.1183867
  Male   0.6804195 0.3195805
```

With the condition of being female or male the revenues are different