BAYESIAN NETWORK ON RNASEQ DATA:

Abstract:

In this paper, it is portrayed the importance of casual effects that emphasizes on cause-effect relationship and experimental design. Correlation does not imply causality and the conclusion merely based on correlation design causes spurious results or Simpson's paradox that leads to the inaccurate and rigorous statistical analysis due to bias. It is proved in the example that using the causal effects can determine the most important features, affecting the target variable, perform more accurate feature selection that leads to the improvement of the accuracy and metrics performance.

Keyword: Correlation, Causal structure learning, PC-simple algorithm, Metrics Performance

1-Experimental Design, Correlation Design and Simpson's Paradox:

The correlation and causality are different. Correlation does not equal causation. When two variables are correlated, that does not mean that one is causing the other.

1-1) Correlation Design: Correlation is focused on observational data, observed as they occur naturally. As for the correlation, we need more variables (third variable or more) to help us conclude cause and effect relationship. The example of correlational design is whether smoking cause cancer as these two variables have high Pearson correlation. It is not sure that smoking causes cancer as there could be other variables that also could develop cancer such as hereditary genes, anxieties or lack of exercise and diet. That is why we need causality.

1-1-2) Problems of Correlation: These problems are: 1) Correlation does not assume the cause-and-effect relationship between variables. Correlation does not say which variable is affecting another even if variables are correlated. 2) Some correlations are spurious. For example, ice cream sells have a high correlation with crime rates or increasing forest fires and increasing death by drownings. 3) Correlation is easy proving causation is hard. Correlation is a vital part of helping us move to the next step the discovery of causation. 4) Correlation Design could cause Simpson's paradox.

1-1-3) Simpson's Paradox: This happens in the correlation design when the attention is on the data instead of the context of data. It is when a correlation appears in several subsets of the data but disappear or reverse when the subsets are combined. For example, in the
Table 1 and
Figure 1, when looking at the treatment for small kidney stones and larger kidney stones individually, it is shown that treatment A is more successful than treatment B. Figure 1 shows individual and combination of treatment A and treatment B. However, when the results are combined for both, treatment B is more successful. This is because the population of patients with small kidney stones, that are much more than population of patients with large kidney stones, received treatment B. That caused the bias on the overall/combined conclusion of treatment B results. As a result, Simpson's paradox leads to inaccurate conclusions about the

causal links between variables that causes bias decision-making based on complex data. These correlation problems require the researchers to consider experimental design.

	Treatment A	**Treatment B**
Small Kidney Stones	93% (81/87)	87% (234/270)
Large Kidney Stones	73% (192/263)	69% (55/80)
Both	**78% (273/350)**	**83% (289/350)**

Table 1: Simpson's Paradox of Treatment A and Treatment B on patients with Small and Large Kidney Stones.

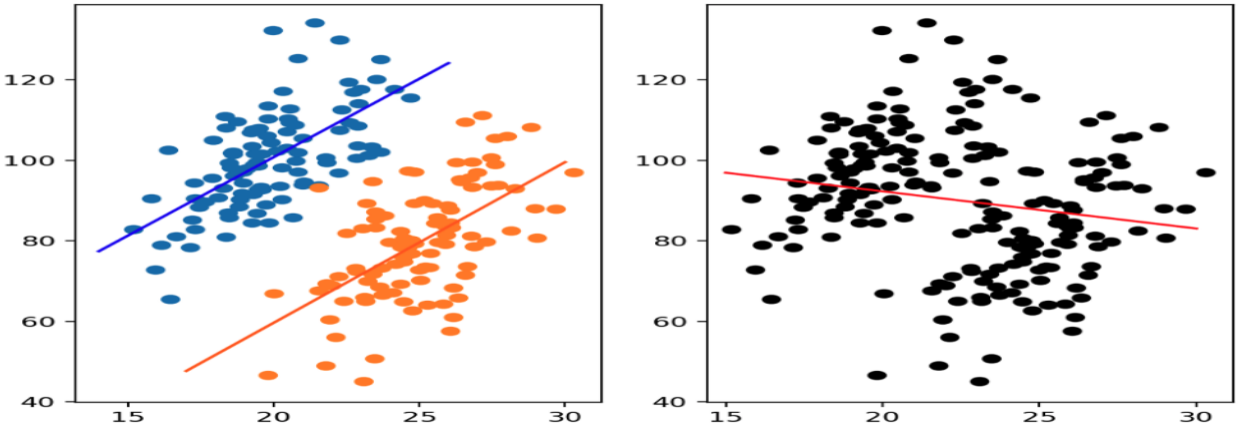

Figure 1: Left: Simpson Paradox on Individual Treatment B and A Where Treatment B is With Blue Color and Treatment A is With Red Color. Right: Is combined Treatment A and Treatment B

1-3) Intervention Experimental Design: Causation is actual cause-effect relationship when it is claimed that one thing causes another thing to happen. In an experimental design, the researcher manipulates the independent variables of X and measure its effects on a dependent variable of Y. For example, in the pharmaceutical industry to make claim that medicine causes a certain effect such as lowering blood pressure or cholesterol, the FDA require companies to do experimental design. This requires putting the medicine through 4 phases that uses control groups and clinical trials to test medicine and make sure X causes Y with more than 95% accuracy. This may take 15 years for the medicine to pass all those phases.

1-4) Conclusion of Correlation and Experimental Design: However, sometimes it is unethical or expensive to only rely on the experimental design. For example, it is very unethical to healthy individuals as control variables of smoking experiments to determine whether smoking causes lung cancer. It is also not feasible and is expensive to do experiments for the gene expressions. Therefore, the combination of correlation design and experimental design algorithms are used.

2)Methodological Consideration of Causal Inference: The method of causal inferences is to use PC-simple algorithm on correlation design to demonstrate the causality. It is assumed that we have correlation variables on the observation data. PC-simple is applied on the correlation design for feature selection and determination of the most important features, affecting target variables that lead to the improvement of matrix performance. The steps that how PC-simple algorithm works is explained as follows:

2-1) How PC-simple works: PC-simple (Neapolitan & Jiang, 2009) is a simplified version of the PC algorithm and follows the same causal assumptions as those by the causal Bayesian network. This algorithm produces a set of variables that strongly influence target variables. There are j different predicted variables in the data sets that are X (1), X (2) to X(j), and the target variable is Z. The goal is to find the parental children set of target variable Z. First, it is assumed that all the predicted variables are in the parents and children set (PC) that have correlation with each other. Then the conditional independence test is applied to test the dependency between each predicted variable and variable target condition on S as it is indicated in $S \in PCK/\{Y\}$ Equation 1). As $S \in PCK/\{Y\}$ Equation 1 indicates, S includes all PC members in level (K) except for the pair (Y) of the target variable.

$$S \in PCK/\{Y\} \qquad \textit{Equation 1}$$

The different steps of PC-Simple algorithm in the Level 1, Level 2, Level 3, and Level 4 are detailed as follows. These different steps of PC-Simple algorithm are shown on Figure 2 and assume Z is a target variable.

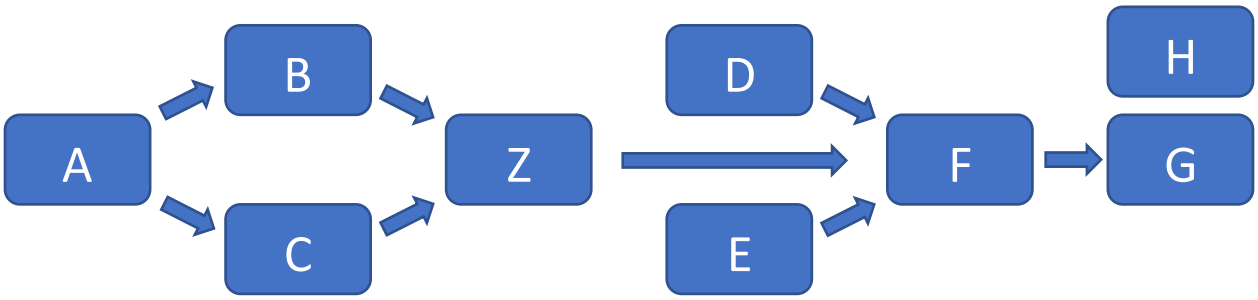

Figure 2: PC-Simple Graph Before Applying some Independence Test that Leads to the Removal of Connection Between Nodes

Level 1: In the first level, the independence condition is on the empty given set (|S|=0). Initially all predicted Variables (PC0= {A, B, C, D, E, F, G, H}) are considered as children and parent set. Examine the independence test of each variable with target variable Z given empty set S. At the end of this iteration, D, E and H are removed from PC1 since they are independent of Z, and PC updates to PC1 = {A, B, C, F, G}.

Level 2: In the second level, the independence condition is on the S with size one (|S|=1).

In this level, we examine the independency condition test of each variable in PC1 with the target variable condition on S. At this level, the size of S is 1, which contains all single variables of PC1 except for the target pair. We assume node G is independent of Z given F, so G is removed from PC1 and the set update to PC2= {A, B, C, F}.

Level 3: In the third level, the independence condition is on the S with size two (|S|=2).
At this level, S implies all members of PC2 with size two except for the target pair. For example, we check the dependency of A with Z given {B, C}, {B, F} and {C, F}. Assuming A is independent of Z given {B, C}, we remove A from PC2 and update the set to PC3= {B, C, F}. The algorithm will be stopped if the size of the last updated PC (PCk) is smaller or equal to the level. In this example size of PC3 is smaller than level 4 (3<4), so it stops, and PC3 is a final set for parents and children of Z. PC-Simple by following graph and assume Z is a target variable.

2-2) Practical Example of Causal Discovery: In this paper, BRCA-50 that is a Breast cancer dataset that include the expression levels of 50 important genes in Breast cancer is studied. The dataset includes 1212 samples with 112 samples are of normal cases (class = N) and 1100 samples are of cancer patients (class = C). In this example, it is determined to accomplish the following goals through the causal inferences.

2-2-1) Gene regulatory network

Causal structure learning algorithm is used to find the gene regulatory network, i.e., the network showing the interactions between genes, using the gene expression data. The Implementation 2-2-1): Gene Regulatory **Network** is as follows.

Implementation 2-2-1): Gene Regulatory Network

To find the best causal structure, constraint-based approach is applied. After replacing the value of the Class variable with 0 and 1, PC arguments is regulated. In the following script, by suffStat, a list of variables from data that have correlations with each other are specified. Then, a conditional independence test will be applied to this list. The conditional independence test of indepTest defines a function to test conditional independence in a significance level of alpha. Furthermore, by labels, we identify a vector of variable names for the algorithm. The output demonstrates the DAG for all variables.

```
knitr::opts_chunk$set(echo = TRUE)
library(pcalg)
data <- read.csv("./BRCA_RNASeqv2_top50.csv")
data$class <- ifelse(data$class =="C", 1, 0) data$class <- as.numeric(data$class) genes <-
subset(data, select=-(class))
n <- nrow (genes)
V <- colnames(genes)
pc.fit <- pc(suffStat = list(C = cor(genes), n = n), indepTest = gaussCItest, alpha=0.01,
labels = V)
if (require(Rgraphviz)) {
plot(pc.fit , main = "Graph 1 - Causal Structure") }
```

Figure 3: Implementation of Gene Regulatory Network with R Statistical Software

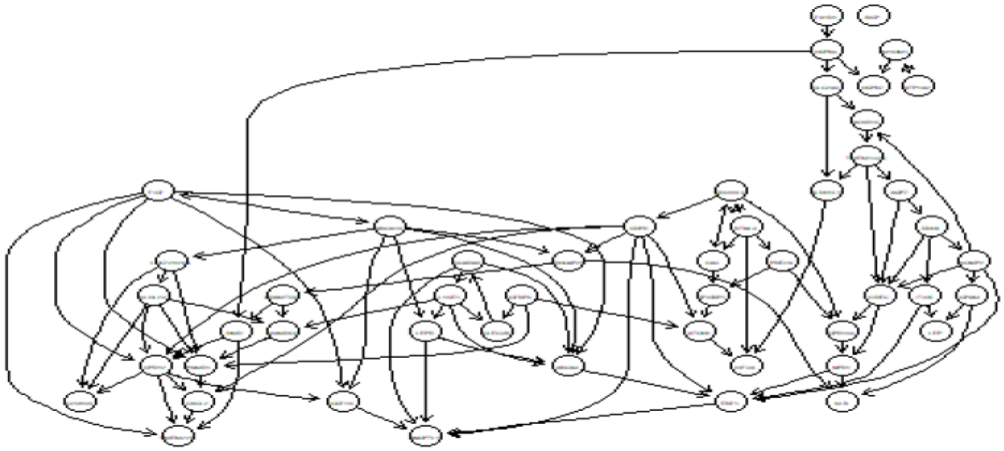

Figure 4: Output of Gene Regulatory Network to Find Gene Regulatory Network that Shows the Interaction Between Genes, Using the Gene Expression.

2-2-1)- Determination of the Most Important Features, Affecting the EBF1 (i.e., important gene):

To find parent and child nodes of the EBF1, we need a local structure learning algorithm through PC-simple. The target/response vector of EBF1 as a first argument has been set. Then all variables except for the response variable were selected for the second argument, and a 0.05 significant level was identified for alpha. The output shows the sorted list of 10 genes that have a strong causal effect on EBF1. The implementation of the above argument, using R statistical software is as follows.

Implementation 2-2-1) 10 Genes with the Strong Causal Effects on Target Variable (EBF1)

The following output shows the list of genes obtained by PC-simple algorithm on discretized data set of gene.

```
pcS <- pcSelect(genes[c("EBF1")], genes[, !names(genes) %in% c("EBF1")] , alpha=0.05)
pcs_frame <- data.frame(pcS,stringsAsFactors = FALSE)
pcs_frame$gene <- rownames(pcs_frame)
rownames(pcs_frame) <- NULL
newdata <- pcs_frame[order(-pcs_frame$zMin),]
newdata[0:10,]
## G zMin gene
## 22 TRUE 7.454331 ABCA9
## 9 TRUE 6.091818 KCNIP2
## 49 TRUE 4.458296 ANGPTL1
## 19 TRUE 3.377566 ARHGAP20
## 20 TRUE 2.867472 NPR1
## 33 TRUE 2.852502 ITIH5
## 6 TRUE 2.772298 SDPR
## 3 FALSE 1.955257 CD300LG
```

Figure 5: Using PC-Simple Algorithms to Find Genes that Have Strong Causal Effects on Target Variable

2-2-2) Feature Selection and Improvement of Metrics Performance

Causal feature selection can improve the performance metrics than the performance of model on all features. The steps of using this method are as follows

Step 1) Applied feature selection based on using PC-simple algorithm to find parent and children set of the class variable (Step 1) **Parent and children set of the class variables by PC-simple** .

Step 2) Evaluated the accuracy of Naïve Bayes classification for all features (Step 2-2) **Naïve Bayes classification with All Features**and selected features (Step 2-1) **Naïve Bayes classification with Selected Features** in the parent and children set of the class variable.

Step 3) Created the performance matrix for the selected features (Step 3-1) **Performance Metrics on Selected FeaturesStep 3-1) Performance Metrics on Selected Features**and all features (Step 3-2) **Performance Metrics on All Features**

Step 4) Compared the results of selected features and all features together, using the 10-fold cross validation (Step 4) Comparison the Results of Performance Metrics on Selected Features and All Features The Implementation **2-2-2)**of these steps are as follows:

Implementation 2-2-2)
Step 1) Parent and children set of the class variables by PC-simple

```
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
library("caret")
library(e1071)
library("klaR")

mean_all <- mean(as.matrix(genes))
genes <- as.data.frame(ifelse(genes > mean_all, 1, 0))
pcS4 <- pcSelect(data[c("class")], genes , alpha=0.05)
pcs4_frame <- data.frame(pcS4,stringsAsFactors = FALSE)
pcs4_frame$gene <- rownames(pcs4_frame)
rownames(pcs4_frame) <- NULL
newdata4 <- pcs4_frame[order(-pcs4_frame$zMin),]
pcs_genes <- newdata4[newdata4$G==TRUE,c("gene")]
pcs_genes
##  [1] "FIGF"     "ARHGAP20" "CD300LG"  "CXCL2"    "KLHL29"   "ATP1A2"
##  [7] "TMEM220"  "MAMDC2"   "SCARA5"   "ATOH8"    "C2orf40"
```

Figure 6: Implementation of PC-Simple to Identify the Parent and Children Set of Class Variables

Step 2-1) Naïve Bayes classification with Selected Features
The train function from the caret library is applied to classify data based on the Naive Bayes approach. First, data is split into train and test and then assign the train set features to the first argument and the outcome for each sample to the second argument. The Naive Bayes as a string

(nb) implies which classification model to use, evaluate and compare the learning algorithm. The trainControl and set cross-validation (cv) is applied as a method. Cross-validation is a resampling method in which the number of folds is the number of groups a given data sample will be split into. The following confusion matrix is obtained based on the output of Naïve Bayes classification on genes by PC-simple algorithm and 10-fold cross validation.

```
genes_select <- genes[,c(pcs_genes)]
train_index <- sample(1:nrow(genes_select), 0.8 * nrow(genes_select))
test_index <- setdiff(1:nrow(genes_select), train_index)
train <- genes_select[train_index,]
test <- genes_select[test_index,]
x <- train
y <- data[train_index,]$class
model = train(x,y,'nb',trControl=trainControl(method='cv',number=10))
x <- test
y <- data[test_index,]$class
confusionMatrix_select <- prop.table(table(predict(model$finalModel,x)$class,y))
confusionMatrix_select
##    y
##           0       1
##   0 0.07407407 0.00000000
##   1 0.02469136 0.90123457
```

Figure 7: Implementation of Naïve Bayes Classification With Selected Features

Step 3-1) Performance Metrics on Selected Features

The following output represent performance metrics for the Naïve Bayes classification on features obtained by PC-simple algorithm.

```
accuracy <- (confusionMatrix_select[1,1]+confusionMatrix_select[2,2])/
  (confusionMatrix_select[1,1]+confusionMatrix_select[1,2]+
     confusionMatrix_select[2,1]+confusionMatrix_select[2,2])
recall <-  confusionMatrix_select[1,1]/
  (confusionMatrix_select[1,1]+confusionMatrix_select[1,2])
Precision <-  confusionMatrix_select[1,1]/
  (confusionMatrix_select[1,1]+confusionMatrix_select[2,1])
F1 <- 2*(Precision*recall)/(Precision+recall)
accuracy
## [1] 0.9753086
recall
## [1] 1
Precision
## [1] 0.75
F1
## [1] 0.8571429
```

Figure 8: Created Performance Metrics For All the Features

Step 2-2) Naïve Bayes classification with All Features

The following confusion matrix based on the output of Naïve Bayes classification on all genes in the data set by applying 10-fold cross validation.

```
data$class <- as.factor(data$class)
train_index <- sample(1:nrow(genes), 0.8 * nrow(genes))
test_index <- setdiff(1:nrow(genes), train_index)
train <- genes[train_index,]
test <- genes[test_index,]
x <- train
y <- data[train_index,]$class

model = train(x,y,'nb',trControl=trainControl(method='cv',number=10))
x <- test
y <- data[test_index,]$class
confusionMatrix <- prop.table(table(predict(model$finalModel,x)$class,y))
confusionMatrix
##    y
##             0         1
##   0 0.06995885 0.00000000
##   1 0.03292181 0.89711934
```

Figure 9:

Step 3-2) Performance Metrics on All Features

The following output represent performance metrics for the Naïve Bayes classification on all features.

```
accuracy <- (confusionMatrix[1,1]+confusionMatrix[2,2])/
(confusionMatrix[1,1]+confusionMatrix[1,2]+ confusionMatrix[2,1]+confusionMatrix[2,2])
recall <- confusionMatrix[1,1]/ (confusionMatrix[1,1]+confusionMatrix[1,2]) Precision <-
confusionMatrix[1,1]/ (confusionMatrix[1,1]+confusionMatrix[2,1])
F1 <- 2*(Precision*recall)/(Precision+recall)
accuracy
## [1] 0.9670782
recall
## [1] 1
Precision
## [1] 0.68
F1
## [1] 0.8095238
```

Figure 10:

Step 4) Comparison the Results of Performance Metrics on Selected Features and All Features

	Accuracy	*Precision*	*F1*
All Features	*0.967*	*0.68*	*0.80*
Selected Features by PC-Simple	*0.97*	*0.75*	*0.85*

Table 2: Comparison of Accuracy Performance for All selected Features and All Features

The results of Table 2 shows the improvement of accuracy and performance matrix for selected features of Naïve Bayes classification model obtained by PC-algorithm. It confirms how the selected influential features/variables, that is the aim of local causal structure learning algorithms, can improve the result of classification model.

4)Conclusion:

In this paper, the PC-simple algorithm is used on the correlation design data sets for causal discovery. This causal discovery, using PC-simple algorithm can help select the most important features, affecting the dependent variable. This appropriate feature selection causes the improvement of the accuracy and precision by 7%. However, PC simple is mostly used for CPDAG (partially directed acyclic graphs) data and does not provide the direction in all the parent children set nodes. The Hiton PC algorithm is also used for causal discovery in which it gives direction to all parent child nodes, even the CPDAG graphs that have no direction in the data sets.

Bibliography

Neapolitan, R., & Jiang, X. (2009). Probabilistic Methods for Bioinformatics: With an Introduction to Bayesian Networks. Elsevier Science & Technology.