Development and application of a package implementing TreeScan software in R

Sohee Jeong

The Graduate School
Yonsei University
Department of Biostatistics and Computing

Development and application of a package implementing TreeScan software in R

A Master's Thesis
Submitted to the Department of Biostatistics and Computing
and the Graduate School of Yonsei University
in partial fulfillment of the
requirements for the degree of
Master of Science

Sohee Jeong

December 2020

This certifies that the master's thesis of ***Sohee Jeong*** is approved.

Inkyung Jung: Thesis Supervisor

Chung Mo Nam: Thesis Committee Member #1

Goeun Park: Thesis Committee Member #2

The Graduate School

Yonsei University

December 2020

Contents

List of Tables iii

List of Figures v

Abstract vi

1. Introduction 1

2. Overview of tree-based scan statistic 4

   2.1 Background 4

   2.2 Statistical Inference 7

3. Tree-based scan statistic 9

   3.1 Unconditional and conditional analysis 9

   3.2 Tree-based scan statistic in Poisson model 11

   3.3 Tree-based scan statistic in Bernoulli model 13

   3.4 Tree-temporal scan statistic 14

4. Package development 19

   4.1 TreeScan software 19

   4.2 A rtreescan package 21

4.2.1 Components of the package 22

4.2.2 Advantages of the package 26

5. Application 28

5.1 Data explanation 28

5.2 Results 29

5.2.1 Tree-based scan statistic in Poisson model 29

5.2.2 Tree-based scan statistic in Bernoulli model 36

5.2.3 Tree-temporal scan statistic 40

6. Conclusion and Discussion 43

7. Supplementary 45

Reference 59

국문요약 63

List of Tables

Table 1. Data structure of drug-AE pair 9

Table 2. Mathematical formulas for log-likelihood ratio by model 17

Table 3. Mathematical formulas for various index by model........ 18

Table 4. Information about the variables in the count file........ 20

Table 5. Result of signal detection for levofloxacin in the unconditional Poisson model (no covariate adjustment) 31

Table 6. Result of signal detection for levofloxacin in the conditional Poisson model (no covariate adjustment) 32

Table 7. Result of signal detection for levofloxacin in the unconditional Poisson model (covariate adjustment) 34

Table 8. Result of signal detection for levofloxacin in the conditional Poisson model (covariate adjustment) 35

Table 9. Result of signal detection for levofloxacin in the unconditional Bernoulli model 38

Table 10. Result of signal detection for levofloxacin in the conditional Bernoulli model 39

Table 11. Result of signal detection for levofloxacin in the unconditional Tree-temporal scan statistic 41

Table 12. Result of signal detection for levofloxacin in the conditional Tree-temporal scan statistic 43

List of Figures

Figure 1. Example of a disease diagnosis tree 5

Figure 2. Cross-table of cases and times 16

Figure 3. Help for the rtreescan package (Demonstration version) 21

Figure 4. Plot of entiretree object 25

Figure 5. Plot of signaltree object 25

Figure 6. Plot of signal detection for levofloxacin in the Poisson model (no covariate adjustment) 29

Figure 7. Plot of signal detection for levofloxacin in the unconditional Poisson model (covariate adjustment) 33

Figure 8. Plot of signal detection for levofloxacin in the conditional Poisson model (covariate adjustment) 33

Figure 9. Plot of signals detected for levofloxacin in the unconditional Bernoulli model 36

Figure 10. Plot of signals detected for levofloxacin in the conditional Bernoulli model.. 37

Figure 11. Plot of signals detected for levofloxacin in the conditional Tree-temporal scan statistic 40

Abstract

The tree-based scan statistic proposed by Kulldorff (2003) is a useful data mining method for detecting signals of excessive risk in hierarchical data. In post-market drug safety surveillance (PMS), the method plays an important role as an "early warning system" that can detect drug safety issues more quickly than traditional methods. Adverse events data coded with medical dictionaries such as WHO Adverse Reactions Terminology (WHO-ART) and Medical Dictionary for Regulatory Activities (MedDRA) is effective for this purpose.

TreeScan is software for various versions of the tree-based scan statistic. The software has some inconveniences and limitations due to the graphic user interface (GUI) design. Users need to use another program to pre-process data in the format required by TreeScan. It is also cumbersome to analyze multiple datasets because it takes a long time to manually change input data and options. We introduce a new and convenient R package "rtreescan" that can solve this problem.

This study has two main objectives. The first purpose is to develop an R package "rtreescan" that implements TreeScan software in R. The second is to understand each type of tree-based scan statistic and outline mathematical formulas such as statistics and relative risks by model. We explain the analysis results by applying each method to the KIDS KAERS database (KIDS-KD) reported from 2010 to 2016.

In conclusion, we have developed a package that enables all analysis processes, including data pre-processing, analysis, and visualization, in one program. As simulation studies become possible, it is likely to contribute to the development of the tree-based scan statistic. In addition, for the first time in this study, the formulas for all methods of this statistic have been summarized. This study is expected to help provide an overall understanding of the tree-based scan statistics and facilitate future studies by researchers.

Keywords: Tree-based scan statistic, Adverse events, Post-market drug safety surveillance, Pharmacovigilance, Signal detection

1. Introduction

To protect public health, it is important to detect unsuspected adverse drug events (AEs) after new drugs are marketed. Not all AEs are reported even in large clinical trials due to the short duration of studies, limited sample size, and limited population representativeness, and so on. In post-market drug safety surveillance (PMS), rare but serious AEs that did not occur in pre-approval clinical trials can be found. Therefore, the importance of monitoring drugs after their approval is growing.

PMS is generally based on spontaneous reporting systems (SRSs), which consist of reports from patients, doctors, pharmacists and other healthcare professionals. In the United States of America, the FDA Adverse Event Reporting System (FAERS, formerly AERS), the Vaccine Adverse Event Reporting System (VAERS) are used to collect and assess the safety of drugs and vaccines, respectively. VigiBase is a World Health Organization's (WHO) global Individual Case Safety Report (ICSR), maintained by the Uppsala Monitoring Center on behalf of the WHO. Also, South Korea has the Korea Adverse Events Reporting System (KAERS) managed by the Korea Institute of Drug Safety and Risk Management (KIDS). SRSs have limitations like under-reporting and length of time on the market. Nevertheless, SRS databases can be effective in detecting potentially unusual or rare AEs. Data mining methods have widely been applied to this type of data.

In pharmacovigilance, data mining methods play a crucial role as an “early warning system” that could detect drug safety issues more quickly than traditional methods. The most widely used method is disproportionality analysis based on calculations using a 2×2 contingency table of drugs-AEs. It consists of several frequentist and Bayesian methods. Frequentist methods include proportional reporting ratio (PRR), reporting odds ratio (ROR), likelihood ratio test (LRT)-based method. Bayesian methods include Bayesian confidence propagation neural network (BCPNN), information component (IC), multi-item gamma Poisson shrinker (MGPS), simplified Bayes (sB), among others.

Tree-based scan statistic (Kulldorff., 2003) is another useful data mining method for detecting signals of excessive risks. The method is suitable for hierarchical structure data in which AEs are coded with medical dictionaries such as WHO Adverse Reactions Terminology (WHO-ART) and Medical Dictionary for Regulatory Activities (MedDRA). The tree-based scan statistic finds statistically significant cuts, i.e. signals, by scanning all possible cuts in all branches. The test statistics are calculated based on the likelihood function for each cut. The hypothesis testing is performed with p-values estimated by Monte-Carlo simulation.

TreeScan is free software that implements various versions of tree-based scan statistic (http://www.treescan.org). There are some inconveniences and limitations due to the graphic user interface (GUI) design. First, users need to use another program to pre-process data in the format required by TreeScan. Second, it is also cumbersome to perform analyses repeatedly using different datasets, like a simulation study. Third, users need to understand how to use TreeScan. We introduce a new and convenient “rtreescan” package that can solve this problem.

In Section 2, we provide a brief overview of tree-based scan statistic for overall understanding. In Section 3, we outline the mathematical formulas of the method based on several probability models. we also look at the meaning of conditional and unconditional analysis. Section 4 explains the TreeScan software and the developed R package. The components and advantages of the package are described in detail. In Section 5, we analyze the KIDS KAERS database (KIDS-KD) data. Finally, we discuss our results and provide some conclusions in Section 6.

2. Overview of tree-based scan statistic

In this session, we briefly explain the concept of a tree-based scan statistic. First of all, it is necessary to understand the appropriate usage situation and tree structure. And this session covers the procedure of calculating the statistic and the statistical inference process.

2.1 Background

Tree-based scan statistic (Kulldorff, 2003) is a statistical data mining method used for signal detection in large healthcare databases. Unlike other data mining methods, this method needs only minimal prior assumptions on the granularity of AEs. In addition, only the tree-based scan statistic accurately controlled type I errors in a study comparing several data mining methods for signal detection (Park et al., 2020). Hence, it is useful for detecting unsuspected relationships with disease risk when independent variables are hierarchical structures.

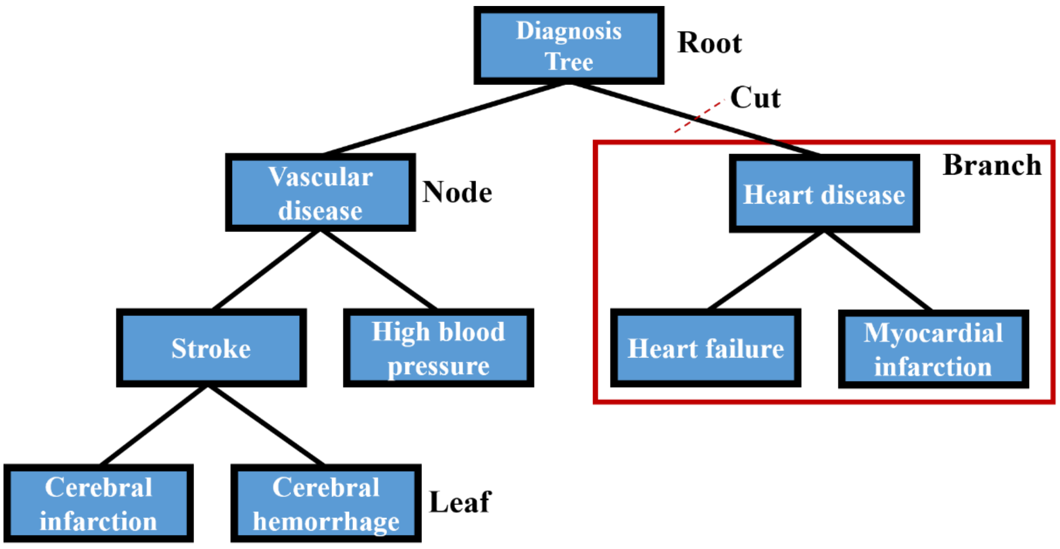


Figure 1. Example of a disease diagnosis tree

Since this method is based on the theory of scan statistics, a hierarchical tree structure pre-specified by the user is required as a scanning window. A tree can be organized into different types depending on the purpose of the study. For example, if the purpose of the study is to detect AEs in a new drug or vaccine, a disease diagnosis tree is used. Figure 1 shows an example of a very small disease diagnosis tree. Conversely, the pharmaceutical drug tree is appropriate for the case we want to know what kind of drug is causing a specific AE. Also, the occupational tree is useful if the interest of the study is finding out which occupational groups have a high risk of getting a certain disease.

The tree is made up of nodes and branches. The branches connecting the leaves represent how closely one node is related to the other. The leaf, which is a node at the bottom of the tree, corresponding to a specific AE. Each leaf contains information about the total number of patients with a specific AE and the number of patients with a specific AE from a certain drug. The related leaves belong to the same higher level, called nodes. A cut is defined as a branch with more observed cases than expected cases, a set of nodes just below the cut.

This method assesses risks simultaneously by scanning for all possible cuts on any branch, adjusting for the multiple testing inherent in the many overlapping groups evaluated. The test statistic is defined as the maximum value of the log-likelihood ratio test statistic comparing a potential cut versus the remaining areas. The cut with the maximum log-likelihood ratio (LLR) is called the Most Likely Cut (MLC). Monte Carlo simulation is used to evaluate the statistical significance of detected cuts. The key is that statistically significant signals do not imply causality. In contrast to most epidemiological studies that evaluate predefined hypotheses, data mining aims to monitor many unsuspected relationships simultaneously. This make it possible to take subsequent action promptly by detecting signals.

2.2 Statistical Inference

Most scan-based statistics are known to be difficult to derive theoretical distributions for statistical significance tests. Since the null distribution of test statistic T is generally unknown, Monte Carlo hypothesis testing is conducted. This procedure is to build an empirical distribution of the test statistic T in order to evaluate its statistical significance (Dwass, 1957).

To calculate the p-value of detected cuts, random datasets are generated multiple times under the null hypothesis. The number of random datasets is usually used as a number ending in 999 such as 999 or 9999. Test statistics are calculated for each random dataset as well as for the real dataset. The Monte Carlo p-value is obtained as $\frac{R}{(M+1)}$, where R is the rank of the test statistic from the real dataset and M is the number of randomly generated datasets. This means comparing the maximum likelihood ratio of MLC in the real dataset and the maximum likelihood ratio of MLCs in the random datasets. A high R ranking is an evidence to reject the null hypothesis. If R is among the 5 percent highest, then the test is significant at $\alpha = 0.05$ level of statistical significance.

In TreeScan, M is set to 999 to provide superior power regardless of data type. For small to medium-size datasets where computing time is not a problem, M is recommended at least 9999 times.

A summary of the procedure for calculating the tree-based scan statistic is as follows:

Calculation procedures of the tree-based scan statistic
1. Scan the tree by considering all possible cuts on any branch.
2. For each cut, calculate the likelihood.
3. Denote the cut with the maximum likelihood as the MLC.
4. Under the null hypothesis, generate *M* (999 or 9999) Monte Carlo replications.
5. Compare the MLC from the real dataset with the MLCs from the random datasets.
6. If the rank of the MLC from the real dataset is *R*, the p-value for that cut is $\frac{R}{(M+1)}$.

3. Tree-based scan statistic

The tree-based scan statistic has been developed in three main types: Poisson, Bernoulli, Tree-Temporal. Each model has a different formula depending on the conditional type. This section describes the tree-based scan statistic based on different probability models.

3.1 Unconditional and conditional analysis

Table 1. Data structure of drug-AE pair

	AE_1	AE_2	...	AE_k	
Case	n_{11}	n_{12}	...	n_{1k}	$n_{1\bullet}$
Control (Bernoulli) or Population (Poisson)	n_{21}	n_{22}	...	n_{2k}	$n_{2\bullet}$
	$n_{\bullet 1}$	$n_{\bullet 2}$	...	$n_{\bullet k}$	

Unconditional and conditional versions depend on whether they are conditioned on the observed data. Literally, conditional scan statistics are conditioned on the total number of observed cases in the tree, but unconditional scan statistics are not. Therefore, in the conditional version, the probability of being a case is replaced as the proportion of cases in the data i.e. $p = n_{1\bullet}/(n_{1\bullet} + n_{2\bullet})$. The unconditional scan statistics need external information about the true probability of being a case under the null hypothesis.

In terms of PMS, it is generally difficult to know the true probability of AEs occurring by a specific drug. The reason is that not all cases are reported even if AEs have occurred. That is, the row margins in Table 1 (row totals) are fixed. we can use conditional scan statistic in this case.

On the other hand, we apply the unconditional scan statistic in self-controlled data and matched data. In these types of data, the true probability of being a case can be assumed. The following is a simple example of a self-controlled design, such as a crossover design. In a vaccine study, cases and controls could be a health outcome occurring 1-14, 29-56 days after vaccination, respectively. In this situation, the probability of being a case is 1/3, since the control risk interval is twice as long as the case risk interval. Another example is the case of using the propensity score matching method to adjust confounders. If the matching ratio of case and control is 1:1, the probability of being a case is set at 1/2.

There is no difference in statistics according to self-controlled design. However, the formulas for relative risk, excess number of cases, and attributable risk are different. In addition, the generated random datasets are affected by the conditional type in the Monte Carlo simulation step. With conditional analysis, the total number of cases in each random dataset is exactly the same as the real dataset. In unconditional analysis, random data is generated either from expected counts (Poisson model) or by using the true probability of being a case (Bernoulli model). The expected counts and the event probability must be pre-specified by the user.

3.2 Tree-based scan statistic in Poisson model

Kulldorff proposed Poisson-type tree-based scan statistic for count data. This is used to model the incidence of AEs. The method is proper if the population can reflect the risk density (e.g. total person-years or covariate-adjusted expected counts).

In mathematical notation, let c_i be the observed number of AEs potentially occurred by a specific drug in leaf i. c_i is approximately Poisson distributed with mean $\lambda_i n_i$, where λ_i is the probability of the ith AE occurring by a specific drug. n_i is the total number of observed AEs in leaf i. $C = \sum_i c_i$ and $N = \sum_i n_i$ are summations for all leaves on the tree. If cut *G* is a group of nearby and related leaves, let $c_G = \sum_{i \in G} c_i$ and $n_G = \sum_{i \in G} n_i$ for each cut *G*.

The hypothesis is as follows:

$$H_0: \lambda_G = \lambda_0 \text{ (for all } i \in G) \;\; vs. \;\; H_a : \lambda_G > \lambda_R \text{ (for at least one } i \in G).$$

Under the null hypothesis, the incidence is the same inside and outside the cut *G*. The alternative hypothesis is that there is at least one cut with a higher incidence than outside the cut. That is, there is at least one leaf or branch with more expected AEs than defined under H_0. However, the null hypothesis can be expressed differently depending on the unconditional and conditional version.

The most likely cut is the node *G* with maximum LLR, test statistic *T* is defined as

$$T = \max_{\mathrm{G}} LLR(G)$$

In the unconditional version, the log-likelihood ratio for cut *G* is as follows.

$$LLR(G) = \left(n_G - c_G + ln\left(\frac{c_G}{n_G}\right)^{c_G}\right) \times I(c_G > n_G)$$

where *I()* is the indicator function, 1 if the observed case of cut G is greater than expected. The indicator function reflects our interest in the study. It can be replaced by $I(c_G < n_G)$ when the alternative hypothesis is $\lambda_G < \lambda_R$.

For the unconditional version, the population means the expected counts under H_0. This version of the statistic focuses on whether the observed count is greater than the expected count for each cut, rather than comparing the incidence rates inside and outside the cut. Since the statistics are not conditioned on the total cases *N*, the null hypothesis is that AEs are generated from the expected counts.

In the conditional version, there is a difference in the formula.

$$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{C - c_G}{N - n_G}\right)^{C-c_G}}{\left(\frac{C}{N}\right)^{C}}\right) \times I(\frac{c_G}{n_G} > \frac{C - c_G}{N - n_G})$$

where *I()* is the indicator function, 1 if there are more events inside than outside the cut *G*.

In a conditional Poisson model, the population can be raw population numbers, a covariate-adjusted population at risk, the expected cases under H_0. The true probability is considered C/N since statistics are conditioned on the total cases *N*. Therefore, the null hypothesis is that the relative risk (RR) of each AE is determined by the relative magnitude of the expected counts, but the total number of AEs is fixed.

3.3 Tree-based scan statistic in Bernoulli model

The Bernoulli model is used for binary data, including disease status, exposure status, etc. Since it models cases that have already occurred at a point in time, it identifies prevalence. Let c_G and n_G are the number of cases in the exposure and the total number of AEs for a given G, respectively. $n_G - c_G$ represents the number of cases in the control.

The hypothesis is as follows:

$$H_0\colon p_G = p_0 \text{ (for all } i \in G) \; vs. \; H_a : p_G > p_R \text{ (for at least one } i \in G).$$

In the unconditional version, the log-likelihood ratio for cut *G* is as follows.

$$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}}{p^{c_G}(1-p)^{n_G - c_G}}\right) \times I\left(\frac{c_G}{n_G} > p\right)$$

where $p = p_0$, which is the probability of being a case under H_0, default to $p_0 = 0.5$. In this case, the null hypothesis is that the probability of an event occurring in the exposure group versus the control group is proportional to p_0. p must be set by the user.

In the conditional version, the formula is defined as

$$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}\left(\frac{C - c_G}{N - n_G}\right)^{C - c_G}\left(\frac{N - n_G - (C - c_G)}{N - n_G}\right)^{N - n_G - (C - c_G)}}{\left(\frac{C}{N}\right)^{C}\left(\frac{N - C}{N}\right)^{N - C}}\right) \times I\left(\frac{c_G}{n_G} > \frac{C - c_G}{N - n_G}\right)$$

The null hypothesis in the model is that the probability of an event occurring in the exposure group versus the control group is proportional to the total number of AEs.

3.4 Tree-Temporal scan statistic

The tree-temporal scan statistic is a combination of the standard tree-based scan statistic and the temporal scan statistic. It is a way to identify how cases are distributed over time in a tree. This requires time information about drug administration and AEs occurrence. It is commonly used in vaccine studies where vaccination times are precisely defined.

Only the tree-temporal scan statistic needs to specify the data time range (start and end range), and study period (i.e. follow-up period, observation period). The start and end ranges are defined as the earliest and last time that data was collected, respectively. For instance, with a study period of 30 days, AEs could occur at 1-14 days and end at 2-25 days. In this case, all temporal clusters with a maximum interval of 15 days (half of the study period), including [1-2], [2], [3-5], and [14-25], are evaluated. This means that all potential time intervals (i.e. risk windows) are evaluated each branch. When assessing a risk window, it is compared with a period that is not within that period, as if the risk window is 14-25 days, the control period is 1-13 plus 26-30 days.

This method is basically conditioned on the number of cases observed in each node. Unconditional and conditional are classified according to whether it is conditioned on time. Namely, unconditional statistic is not conditioned on time. Therefore, it assumes that cases are occur the same probability in study period.

In this model, let c_G is the number of cases in node G that is also in a specific time interval. n_G is the number of cases in node G and $n_G - c_G$ means the number of cases on node G that are NOT in the time interval, respectively.

The hypothesis is as follows:

$$H_0 : \lambda_G = \lambda_0 \text{ (for all } i \in G \text{ over the entire tree)}$$

$$vs.\ H_a : \lambda_G > \lambda_R \text{ (for at least one } i \in G \text{ during some time interval).}$$

In the unconditional version, the log likelihood ratio formula for cut *G* is as follows.

$$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}}{\left(\frac{w}{T}\right)^{c_G}\left(\frac{T - w}{T}\right)^{n_G - c_G}}\right) \times I\left(\frac{c_G}{n_G} > \frac{w}{T}\right)$$

where w : Length of the time interval, T : Length of the total study period

In the model that is unconditional with respect to time, the null hypothesis is that cases occur uniformly over the study period, with equal probability on each day. That is, AEs occur in proportion to the length of the time interval relative to the total study period. The probability of an event occurring in a time interval is assumed to be w/T.

In the conditional version, the formula is defined as

$$LLR(G) = ln\left(\left(\frac{c_G}{u_G}\right)^{c_G}\left(\frac{C - c_G}{C - u_G}\right)^{C - c_G}\right) \times I(c_G > u_G)$$

where c_G : Number of cases for node G in the time interval

n_G : Total number of cases in the node G

C : Total number of cases in the tree

Z : Total number of cases in time interval over the tree

$u_G = n_G \times \frac{Z}{C}$: Expected cases in the time interval in node G under H_0

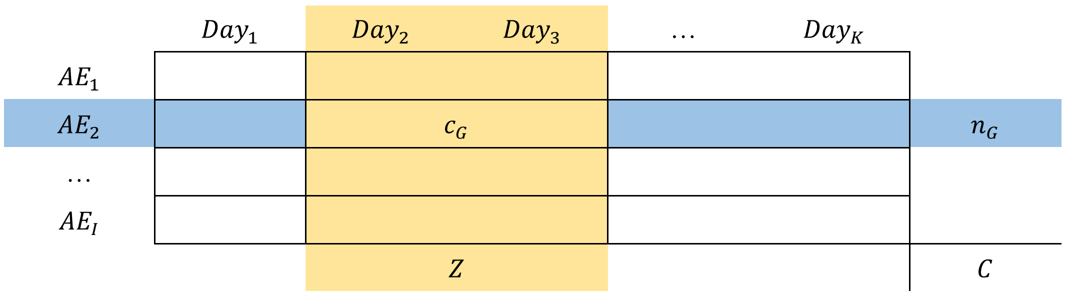


Figure 2. Cross-table of cases and times

The unconditional tree-temporal scan statistic only conditions the number of cases observed at each node, whereas this statistic also conditions the total number of cases occurring in each time unit. Therefore, the probability of an event occurring varies over time. As an example, it is available for cases where the probability of AEs is high early in the vaccination and then lows thereafter.

The null hypothesis is all AEs have the same probability of occurring on a certain time interval. Similar to the conditional Bernoulli model, the probability under H_0 is based on the proportion of the total number of AEs in that time interval and the total number of AEs in the study period (i.e. $\frac{Z}{C}$).

If the specified time interval is 2-3 days and cut G is AE_2, as shown in Figure 2, the expected cases of AE_2 for that period is calculated as $u_G = n_G \times \frac{Z}{C}$. That is, u_G is the part that reflects the variable risk window. In addition, the tree-temporal scan statistic can analyze censored data based on pseudolikelihood. The censoring time can be added to the fourth column of the case file. Please refer to the TreeScan guidelines for details.

Table 2. Mathematical formulas for log-likelihood ratio by model

Type of Scan	Conditional	Log-likelihood ratio
Poisson	No	$LLR(G) = \left(n_G - c_G + ln\left(\frac{c_G}{n_G}\right)^{c_G}\right) \times I(c_G > n_G)$
	Total case	$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{C - c_G}{N - n_G}\right)^{C - c_G}}{\left(\frac{C}{N}\right)^{C}}\right) \times I\left(\frac{c_G}{n_G} > \frac{C - c_G}{N - n_G}\right)$
Bernoulli	No, self-control No, standard	$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}}{p^{c_G}(1 - p)^{n_G}}\right) \times I\left(\frac{c_G}{n_G} > p\right)$
	Total cases	$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}\left(\frac{C - c_G}{N - n_G}\right)^{C - c_G}\left(\frac{N - n_G - (C - c_G)}{N - n_G}\right)^{N - n_G - (C - c_G)}}{\left(\frac{C}{N}\right)^{C}\left(\frac{N - C}{N}\right)^{N - C}}\right) \times I\left(\frac{c_G}{n_G} > \frac{C - c_G}{N - n_G}\right)$
Tree-Temporal	Node	$LLR(G) = ln\left(\frac{\left(\frac{c_G}{n_G}\right)^{c_G}\left(\frac{n_G - c_G}{n_G}\right)^{n_G - c_G}}{\left(\frac{w}{T}\right)^{c_G}\left(\frac{T - w}{T}\right)^{n_G - c_G}}\right) \times I(\frac{c_G}{n_G} > \frac{w}{T})$
	Node and time	$LLR(G) = ln\left(\left(\frac{c_G}{u_G}\right)^{c_G}\left(\frac{C - c_G}{C - u_G}\right)^{C - c_G}\right) \times I(c_G > u_G)$

Table 3. Mathematical formulas for various index by model

Type of Scan	Conditional	Expected	Relative Risk	Excess Cases
Poisson	No	n	$\frac{c}{n}$	$c-n$
	Total case	$n\frac{C}{N}$	$\frac{c/n}{(C-c)/(N-n)}$	$c-n\frac{(C-c)}{(N-n)}$
Bernoulli	No, self-control	np	$\frac{c}{np}$	$c-np$
	No, standard	np	$\frac{c/p}{(n-c)/(1-p)}$	$c-p\frac{(n-c)}{(1-p)}$
	Total cases	$\frac{nC}{N}$	$\frac{c/n}{(C-c)/(N-n)}$	$c-n\frac{(C-c)}{(N-n)}$
Tree-Temporal	Node	$\frac{wn}{T}$	$\frac{c/w}{(n-c)/(T-w)}$	$c-w\frac{(n-c)}{(T-w)}$
	Node and time	$\frac{nz}{C}$	$\frac{c(C-n-z+c)}{(n-c)(z-c)}$	$c-\frac{(n-c)(z-c)}{(C-n-z+c)}$
Purely Temporal	Total cases	$\frac{wC}{T}$	$\frac{c/w}{(C-c)/(T-w)}$	$c-w\frac{(C-c)}{(T-w)}$

4. Package development

This section introduces a new package available in R. A rtreescan package operate based on the TreeScan software. After a briefly review of the TreeScan software, we illustrate components and expected effects of the developed package.

4.1 TreeScan software

TreeScan is a data mining software that allows users to analyze large datasets using various versions of the tree-based scan statistic. It looks for excess risk not only in each individual cell in the database but also in groups of closely related cells. It is not open-source, but it is distributed free of charge on the website (https://www.treescan.org).

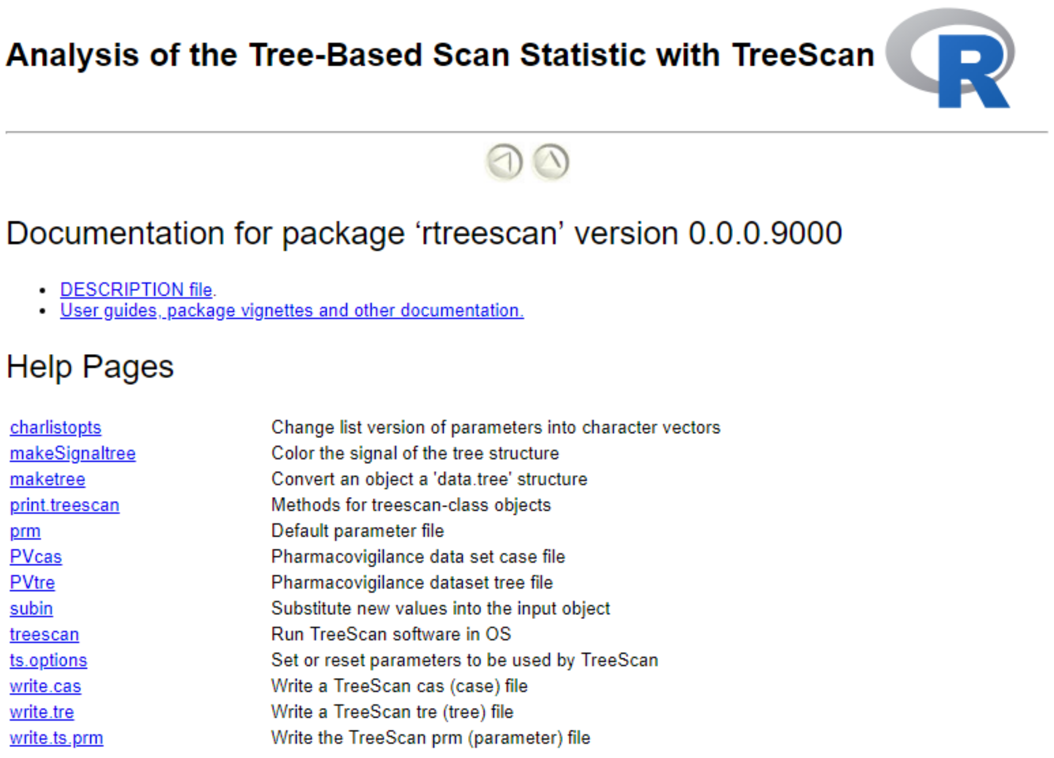
TreeScan requires three files to operate: a parameter file, a tree file, and a count file. The parameter file contains all information about the analysis, including the type of probability model, input and output options. The tree file represents the structure of the tree in two variables: Node ID and Parent Node ID. Every node on the tree is assigned a unique Node ID. The second variable is defined as the parent node above one level of that node. Each row in the tree file represents one node. That is, it has as many rows as the number of nodes including the root. The count file has information about observed case by each node. This file consists of three variables. Basically, it has Node ID and Number of Cases, and the last variable depends on the type of probability model. For the Bernoulli model, the last variable is the number of controls, and for the Poisson model, it is expected cases or total population numbers. The tree-temporal model contains information about case times. Table 4 describes the variables required for the count file.

Table 4. Information about the variables in the count file

Variable	Description
Node ID	Any numerical value or string of characters. Not used for a purely temporal analysis.
Number of Cases	The number of observed cases for the specified node.
Number of Controls (Bernoulli)	The number of observed controls for the specified node.
Population (Poisson)	The population size for the specified node. For the conditional Poisson model, this could be raw population numbers, a covariate adjusted population at risk, or, the expected number of cases under the null hypothesis. If the unconditional Poisson model is used, it must be the expected counts.
Time (Tree-temporal)	Specified in a generic format, which typically represent days, weeks, months or years, although it could also represent seconds, minutes, decades or centuries.

4.2 A rtreescan package

There are two ways of access to TreeScan: a GUI and a batch file. The GUI is easy to perform by direct manipulation of graphic elements but prevents automated and repeated operation. The batch file allows this but can be hard to integrate into other analyses. The rtreescan package based on the batch file enables easy automation and integration. This package allows TreeScan to operate in an R environment. Therefore, TreeScan must be installed in advance to use the rtreescan package.



Analysis of the Tree-Based Scan Statistic with TreeScan

Documentation for package 'rtreescan' version 0.0.0.9000

- DESCRIPTION file.
- User guides, package vignettes and other documentation.

Help Pages

charlistopts	Change list version of parameters into character vectors
makeSignaltree	Color the signal of the tree structure
maketree	Convert an object a 'data.tree' structure
print.treescan	Methods for treescan-class objects
prm	Default parameter file
PVcas	Pharmacovigilance data set case file
PVtre	Pharmacovigilance dataset tree file
subin	Substitute new values into the input object
treescan	Run TreeScan software in OS
ts.options	Set or reset parameters to be used by TreeScan
write.cas	Write a TreeScan cas (case) file
write.tre	Write a TreeScan tre (tree) file
write.ts.prm	Write the TreeScan prm (parameter) file

Figure 3. Help for the rtreescan package (Demonstration version)

4.2.1 Components of the package

The package is composed of R functions, sample datasets, and documents. Sample datasets and documents help users easily use the package. Help illustrates R functions, sample datasets, and other objects. Vignette is a helpful tutorial that guides users on how to use the package.

The R functions of the rtreescan package consist of three main parts: a parameter function to set parameters for TreeScan, an execution function to run TreeScan, and a plotting function to visualize hierarchical tree structures.

Parameter function

The parameter function plays two main roles in setting parameters and writing files to the operating system (OS). First, we need to set parameters for our analysis purposes. we can use ts.options() to check and change the default settings of TreeScan that are set automatically. For example, this function can set the type of probability model, conditional type, input and output options, the number of Monte Carlo replications, and so on.

Second, we must write all input files with a specific file extension to the OS in TreeScan readable formats. This is because TreeScan requires all input files to be in TreeScan ASCII file format. The write.ts.prm() function allow us to save parameter information set by ts.function() to the parameter file with “.prm” extension. In the same way, we can use write.cas() and write tre() to write count file with “.cas” extension and tree file “.tre” extension, respectively.

Execution function

The main function, treescan() runs TreeScan with information pre-specified in the parameter file. This function returns a standard output file with a ".txt" file extension that contains a summary of data, most likely cuts, parameter settings, and computational information. We can also generate additional optional output files in HTML, CSV format by setting output options.

Plotting function

The plotting function is to visualize the analysis results in a tree structure. The maketree() function converts the tree file to data.tree structure required for visualization. The data.tree structure consists of the path text from the root to each leaf. The makeSignaltree() function needs the results of maketree() and treescan() functions. In other words, the maketree() creates an entire tree structure with all leaves connected to their parent nodes. The makeSignaltree() function colors detected signals in the entire tree structure generated by the maketree() function. It is possible to extract statistically significant parts of the tree with the "onlysignal=TRUE" option. This is useful if the tree structure is too large to be visualized. In addition, users can change the style of the nodes or edges to suit their preferences using various functions in data.tree package.

Simulated data

We generated sample datasets (PVcas, PVtre) based on real data for easy use of the packages by users. For this, we used the KIDS KAERS database (KIDS-KD) reported to KAERS between 2012 and 2016. KAERS is designed to report and manage information about abnormal cases after taking medicine and medical supplier. KIDS-KD is coded as WHO-ART for the internationally standardized classification of diseases. It has 4 levels hierarchy that begins with the human organ system. However, PVtre data considered 3 levels only with the terms PT, HT, and SOC. And we extracted a sub-tree with 14 leaves from all the reported leaves from the database. Also, PVcas are simulated with different values, maintaining the pattern of the actual values. Once users load the package, simulated datasets are available in R.

- SOC: System-organ classes body organ groups
- HT: High level terms for grouping preferred terms
- PT: Preferred terms principal terms for describing adverse reactions
- IT: Included terms synonyms to Preferred terms

Vignette guides users with detailed instructions on how to use the package. It is in the supplementary of section 7. Figure 4 on the next page is a visualization of the entire tree structure created by maketree(), and Figure 5 is the result of coloring statistically significant signals using makeSignaltree(arg, onlysignal=TRUE).

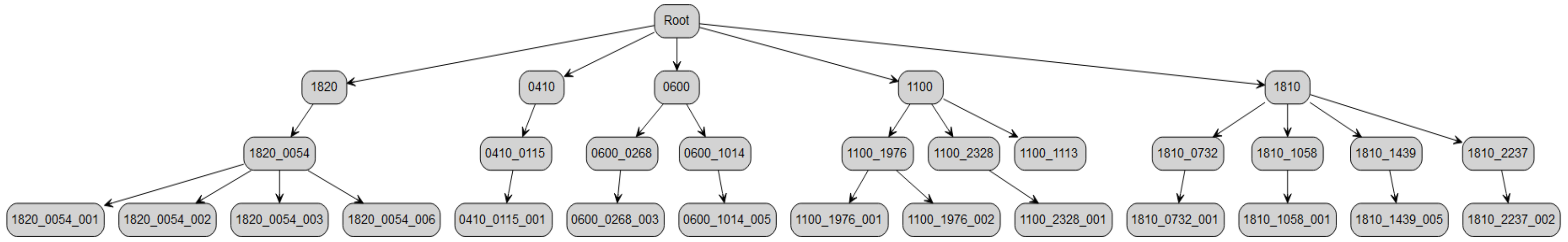


Figure 4. Plot of entiretree object

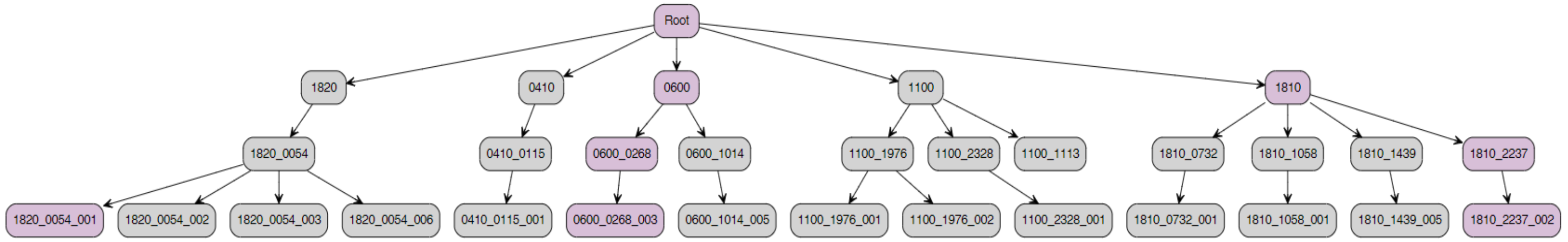


Figure 5. Plot of signaltree object

4.2.2 Advantages of the package

The rtreescan package brings three major advantages to users. First, users can perform all analysis processes in one program. TreeScan requires input files with a certain format. Users need to redefine the data in a format suitable for TreeScan. Users may use another software (e.g. Excel, R) to do this and input the defined data into the TreeScan. This is often cumbersome, especially if users have a lot of data to handle. With rtreescan package, users can preprocess, analyze, and visualize easily in R without multiple software.

Second, this package makes it possible to repeat the analysis using numerous datasets. Not only have countless drugs and vaccines already been developed around the world, but the development of new drugs in the bio-pharmaceutical field has recently become more active. Reported AEs information will also increase as the number of drugs on the market increases. It can be time-consuming to use a GUI-type program whenever Pharmacovigilance is performed on many drugs. If numerous analyses need to be performed like this, it is highly efficient to use this package.

It also enables simulation studies. As mentioned previously, the tree-based scan statistic has been developed recently, and only three types are available. Simulation studies using this package can contribute to active research using the tree-based scan statistic. For instance, we can develop new statistics that complement existing statistics and compare performance with existing ones. Various studies based on simulation studies can be conducted using “rtreescan” package.

Third, users can customize the visualization options of the tree structure. By setting options, users can change the color and shape of the node, the font type, and the thickness of the edge in different styles. When using data with large tree structures, such as real data, it is impossible to visualize the results. In this case, the function to visualize only the detected signals in the package can be useful. Visualized results help users understand the analysis results. Visualization plots help users understand the overall results of the analysis.

5. Application

5.1 Data description

We used the KIDS KAERS database (KIDS-KD) reported to KAERS between 2012 and 2016 to apply the method to real data. The data contains information on demographics, drugs, AEs, reporters, medical history, and a causality assessment. All drugs and AEs are coded by the Anatomical Therapeutic Chemical Classification System (ATC) code and WHO-ART's preferred terms (PTs), respectively.

The frequency table of drugs and AEs includes approximately 1.8 million drug reports with 1,981 types of drugs, and 1.1 million AE reports with 4,078 types of AEs. Only the first report was used for duplicate AE drug pairs reported by the same person at dose or time. We did not include pairs whose causal assessment criteria are lower than 'possible' and whose duration (the length from drug administration to AE occurrence) is less than one day. The final dataset included 360,971 reports with types of 838 AEs. The tree structure consists of SOC and PT term (i.e. two levels), excluding HLT, and IT.

In order to explore safety signals, we considered Levofloxacin, which is used to treat a variety of bacterial infections. In the Bernoulli model, Ciprofloxacin, the same second-generation Quinolones as Levofloxacin, was used as a control.

5.2 Results

The results of signal detection using real data are shown in Figures 6-11 and Tables 5-12. The common AE of Ciprofloxacin (the control group in the Bernoulli model) is similar to Levofloxacin. Levofloxacin-related AEs are reported most frequently for nausea, vomiting, diarrhea, headache, trouble sleeping, abdominal pain, rash, itching, and so on. Ciprofloxacin-related AEs are reported most frequently for nausea, vomiting, diarrhea, headache, abdominal pain, rash, stomach upset (https://www.healthline.com).

Two commonly detected signals (0600_0205, Diarrhoea; 0200_0063; Arthralgia) in the three models are marked in light blue.

5.2.1 Tree-based scan statistic in the Poisson model

The Poisson model performed an analysis of when covariates were adjusted and not adjusted. The adjusted covariates considered age and gender. Age is grouped into 8 categories at 10-year intervals, including under 20s, 20s, and 80s and older. First, we look at the results of not adjusting covariates.

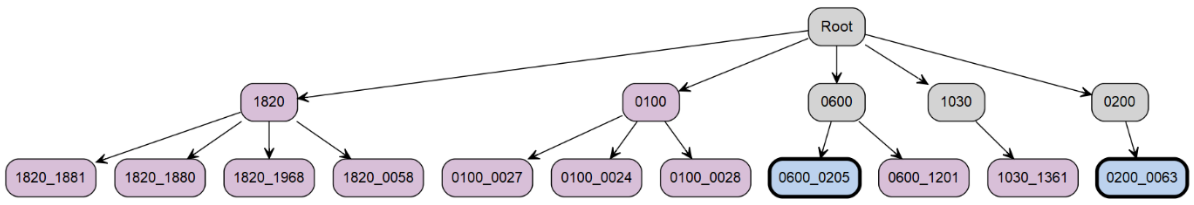


Figure 6. Plot of signal detection for levofloxacin in the Poisson model

(no covariate adjustment)

In this case, the signals detected in the unconditional and conditional versions are the same, as in Figure 6. There is a difference only in RR values in Table 5 and 6.

In this model with 13 signals, AEs on diarrhea and skin disease (0100, 1820) are statistically significantly detected. That is, there is a higher incidence of diarrhea and skin diseases that are potentially caused by Levofloxacin than other drugs. However, it is important to note that this is not a causal relationship where AEs are caused by a particular drug. Significantly detected AEs belong to the Skin and appendages disorders (0100) are as follows: Pruritus (0100_0024), Rash (0100_0027), Rash erythematous (0100_0028). Significantly detected AEs belong to the Application site disorders (1820) are Injection site reaction (1820_0058), Injection site pruritus (1820_1880), Injection site rash (1820_1881), Injection site urticaria (1820_1968). Other detected signals include Arthralgia (0200_0063), Diarrhoea (0600_0205), and so on.

Table 5. Result of signal detection for levofloxacin in the unconditional Poisson model (no covariate adjustment)

Cut No.	Node code	Adverse Event	Obs	Exp	RR	*P*-value
1	0100	Skin and appendages disorders	1174	686.69	1.71	0.001
2	0100_0024	Pruritus	500	253.03	1.98	0.001
3	0100_0027	Rash	400	159.06	2.51	0.001
4	0100_0028	Rash erythematous	23	5.8	3.97	0.001
5	0200_0063	Arthralgia	15	3.19	4.71	0.002
6	0600_0205	Diarrhoea	138	54.16	2.55	0.001
7	0600_1201	Diarrhoea, Clostrdium difficile	7	0.85	8.24	0.016
8	1030_1361	QT prolonged	8	0.49	16.3	0.001
9	1820	Application site disorders	241	38.65	6.24	0.001
10	1820_0058	Injection site reaction	16	5.19	3.08	0.048
11	1820_1880	Injection site pruritus	72	2.74	26.27	0.001
12	1820_1881	Injection site rash	110	7.07	15.56	0.001
13	1820_1968	Injection site urticaria	14	1.08	12.91	0.001

Obs, Observed count; Exp, Expected count; RR, Relative Risk

Table 6. Result of signal detection for levofloxacin in the conditional Poisson model (no covariate adjustment)

Cut No.	Node code	Adverse Event	Obs	Exp	RR	*P*-value
1	100	Skin and appendages disorders	1174	686.69	2.28	0.001
2	0100_0024	Pruritus	500	253.03	2.2	0.001
3	0100_0027	Rash	400	159.06	2.78	0.001
4	0100_0028	Rash erythematous	23	5.8	3.99	0.001
5	0200_0063	Arthralgia	15	3.19	4.73	0.001
6	0600_0205	Diarrhoea	138	54.16	2.63	0.001
7	0600_1201	Diarrhoea, Clostrdium difficile	7	0.85	8.25	0.009
8	1030_1361	QT prolonged	8	0.49	16.34	0.001
9	1820	Application site disorders	241	38.65	6.76	0.001
10	1820_0058	Injection site reaction	16	5.19	3.1	0.034
11	1820_1880	Injection site pruritus	72	2.74	26.98	0.001
12	1820_1881	Injection site rash	110	7.07	16.19	0.001
13	1820_1968	Injection site urticaria	14	1.08	12.97	0.001

Obs, Observed count; Exp, Expected count; RR, Relative Risk

The following are the results of adjusting the covariates. Figure 7 and Table 7 are the results of the unconditional Poisson model and Figure 8 and Table 8 are the results of the conditional Poisson model, respectively.

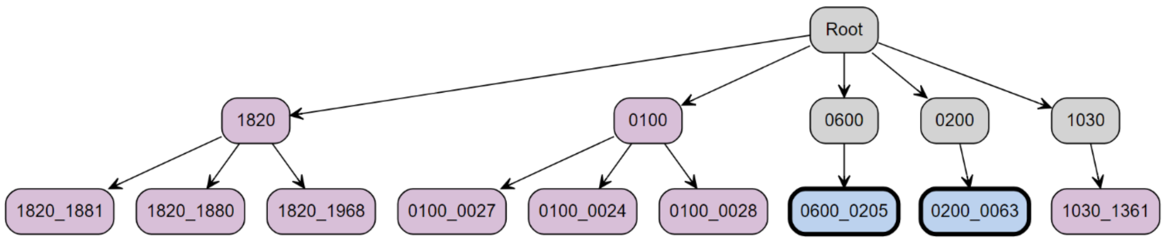


Figure 7. Plot of signal detection for levofloxacin in the unconditional Poisson model (covariate adjustment)

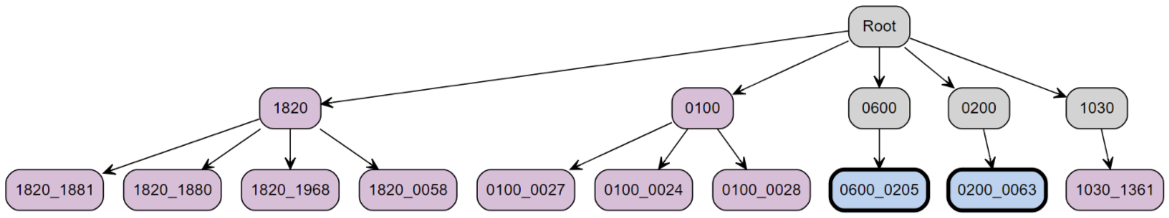


Figure 8. Plot of signal detection for levofloxacin in the conditional Poisson model (covariate adjustment)

In this case, the conditional model detected one more signal of the Injection site reaction (1820_0058) than the conditional model. However, almost identical signals were detected, regardless of whether the model was conditioned or covariate-adjusted. In other words, it can be seen that the occurrence of AEs is not affected by age or gender.

Table 7. Result of signal detection for levofloxacin in the unconditional Poisson model (covariate adjustment)

Cut No.	Node code	Adverse Event	Obs	Exp	RR	*P*-value
1	100	Skin and appendages disorders	1174	661.45	1.77	0.001
2	0100_0024	Pruritus	500	247.48	2.02	0.001
3	0100_0027	Rash	400	156.48	2.56	0.001
4	0100_0028	Rash erythematous	23	5.85	3.93	0.001
5	0200_0063	Arthralgia	15	2.99	5.01	0.001
6	0600_0205	Diarrhoea	138	65.09	2.12	0.001
7	1030_1361	QT prolonged	8	0.72	11.11	0.001
8	1820	Application site disorders	241	37.37	6.45	0.001
9	1820_1880	Injection site pruritus	72	2.83	25.41	0.001
10	1820_1881	Injection site rash	110	6.8	16.17	0.001
11	1820_1968	Injection site urticaria	14	0.96	14.51	0.001

Obs, Observed count; Exp, Expected count; RR, Relative Risk

Table 8. Result of signal detection for levofloxacin in the conditional Poisson model (covariate adjustment)

Cut No.	Node code	Adverse Event	Obs	Exp	RR	*P*-value
1	100	Skin and appendages disorders	1174	661.45	2.39	0.001
2	0100_0024	Pruritus	500	247.48	2.26	0.001
3	0100_0027	Rash	400	156.48	2.83	0.001
4	0100_0028	Rash erythematous	23	5.85	3.96	0.001
5	0200_0063	Arthralgia	15	2.99	5.03	0.001
6	0600_0205	Diarrhoea	138	65.09	2.18	0.001
7	1030_1361	QT prolonged	8	0.72	11.14	0.001
8	1820	Application site disorders	241	37.37	6.99	0.001
9	1820_0058	Injection site reaction	16	5.2	3.09	0.043
10	1820_1880	Injection site pruritus	72	2.83	26.09	0.001
11	1820_1881	Injection site rash	110	6.8	16.83	0.001
12	1820_1968	Injection site urticaria	14	0.96	14.58	0.001

Obs, Observed count; Exp, Expected count; RR, Relative Risk

5.2.2 Tree-based scan statistic in the Bernoulli model

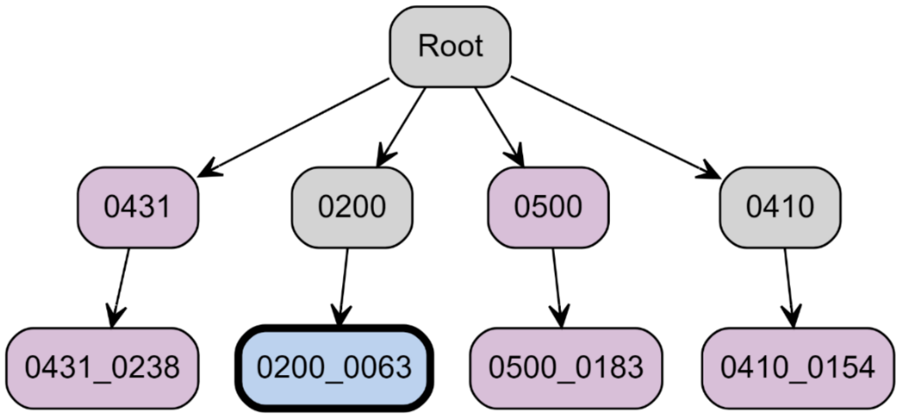


Figure 9. Plot of signals detected for levofloxacin in the unconditional Bernoulli model

The unconditional Bernoulli model used matched data for age and gender. Due to 1:1 matching, the true probability was set to 1/2 and a total of six signals were detected.

AEs were newly detected for Vision disorders (0431), Psychiatric disorders (0500). Because the true probability is set to 0.5, the prevalence of all detected AEs is higher than 0.5. In particular, Conjunctivitis (0431_0238), Insomnia (0500_0183) was detected up to the upper node despite being a single node. It is expected that the number of cases on the child nodes will account for a large proportion of the number of cases on the parent node.

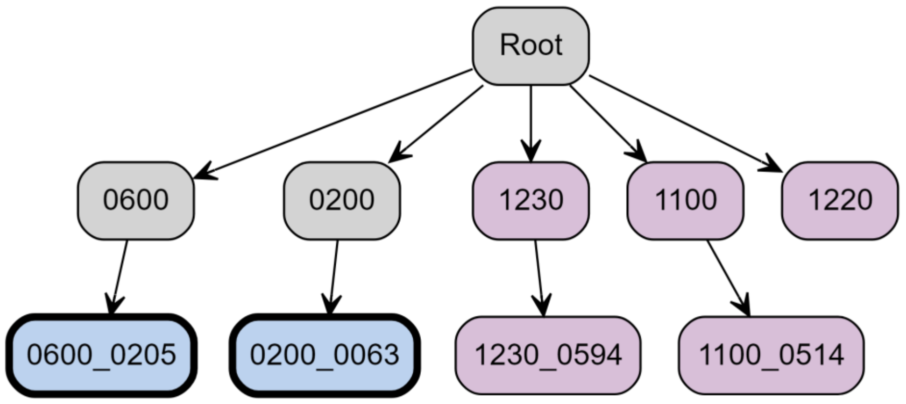


Figure 10. Plot of signals detected for levofloxacin in the conditional Bernoulli model

In the conditional Bernoulli model with 7 signals, AEs were newly detected for Respiratory system disorders (1100), Dyspnoea (1100_0514), White cell and reticuloendothelial system (RES) disorders (1220), Platelet, bleeding & clotting disorders (1230). The detected AE has a higher prevalence of Levofloxacin than Ciprofloxacin. Although it is not common, there is a study that Levofloxacin can cause serious platelet depletion (Polprasert, 2009). Although there is a slight difference in the detected signal depending on the covariate adjustment, Arthralgia (0200_0063) was generally detected.

Table 9. Result of signal detection for levofloxacin in the unconditional Bernoulli model

Cut No.	Node code	Adverse Event	Total Obs	Obs	Exp	RR	*P*-value
1	0200_0063	Arthralgia	40	35	20	1.75	0.001
2	0410_0154	Tremor	33	26	16.5	1.58	0.04
3	0431	Vision disorders	55	50	27.5	1.82	0.001
4	0431_0238	Conjunctivitis	34	33	17	1.94	0.001
5	0500	Psychiatric disorders	152	97	76	1.28	0.038
6	0500_0183	Insomnia	47	37	23.5	1.57	0.006

Total Obs, Total observation (cases and controls); Obs, Observed cases; Exp, Expected cases; RR, Relative Risk

Table 10. Result of signal detection for levofloxacin in the conditional Bernoulli model

Cut No.	Node code	Adverse Event	Total Obs	Obs	Exp	RR	*P*-value
1	0200_0063	Arthralgia	17	15	5.9	2.55	0.001
2	0600_0205	Diarrhoea	243	138	84.34	1.67	0.001
3	1100	Respiratory system disorders	151	72	52.41	1.38	0.044
4	1100_0514	Dyspnoea	105	53	36.44	1.46	0.043
5	1220	White cell and RES disorders	71	38	24.64	1.55	0.048
6	1230	Platelet, bleeding & clotting disorders	27	19	9.37	2.04	0.008
7	1230_0594	Thrombocytopenia	19	14	6.59	2.13	0.028

Total Obs, Total observation (cases and controls); Obs, Observed cases; Exp, Expected cases; RR, Relative Risk

5.2.3 Tree-Temporal scan statistic

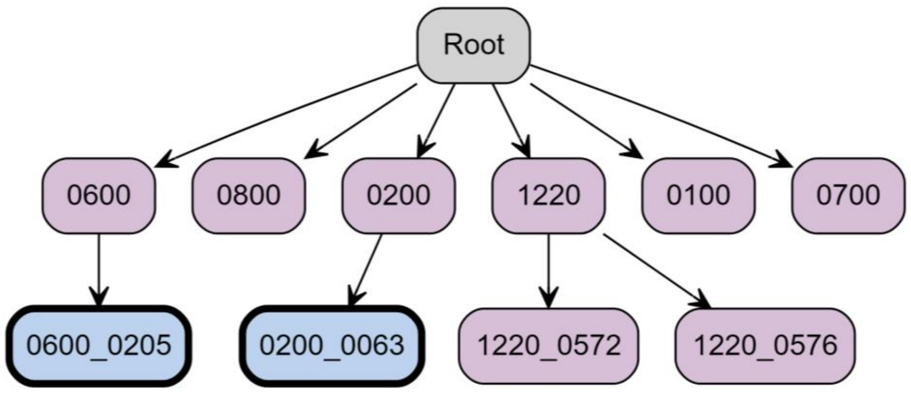


Figure 11. Plot of signals detected for levofloxacin in the conditional Tree-temporal scan statistic

Unconditional tree-temporal scan statistic is not used in general situations, as AEs are assumed to occur uniformly over the study period, with equal probability on each day. As a result of applying the model, 100 signals were detached and therefore visualization was not possible.

In the Conditional tree-temporal scan statistic with 10 signals, the following SOCs have been detected: Skin and appendages disorders (0100), Musculo-skeletal system disorders (0200), Gastro-intestinal system disorders (0600), Liver and biliary system disorders (0700), Metabolic and nutritional disorders (800), and White cell and RES disorders (1220). Among them, Skin and appendages disorders (0100) between 1-2 days after drug administration, Diarrhoea (0600_0205) between 3-20 days drug administration had a higher incidence than the remaining time interval, respectively.

Table 11. Result of signal detection for levofloxacin in the unconditional Tree-temporal scan statistic

Cut No.	Node code	Adverse Event	Obs	Risk Window	Obs in Window	Exp	RR	*P*-value
1	0100	Skin and appendages disorders	1174	1-4	1002	1.37	5002.72	0.001
2	0100_0003	Angioedema	27	1-7	24	0.055	3922.29	0.001
3	0100_0014	Erythema multiforme	2	1-3	2	0.0017	infinity	0.003
4	0100_0024	Pruritus	500	1-4	436	0.58	5850.23	0.001
5	0100_0027	Rash	400	1-8	365	0.93	4472.55	0.001
6	0100_0028	Rash erythematous	23	1-5	22	0.033	15109.6	0.001
7	0100_0036	Skin discoloration	5	1-4	5	0.0058	infinity	0.001
8	0100_0043	Sweating increased	10	1-12	10	0.035	infinity	0.001
9	0100_0044	Urticaria vesiculosa	191	1-5	171	0.28	5872.14	0.001
10	0100_0871	Bullous eruption	4	1-2	3	0.0023	5155.5	0.001
11	0100_1199	Skin exfoliation	2	1-4	2	0.0023	infinity	0.005
…	…		…	…	…	…	…	
100	1820_1968	Injection site urticaria	14	1-2	14	0.0081	infinity	0.001

Obs, Observed count; Exp, Expected count; RR, Relative Risk

Table 12. Result of signal detection for levofloxacin in the conditional Tree-temporal scan statistic

Cut No.	Node code	Adverse Event	Obs	Risk Window	Obs in Window	Exp	RR	*P*-value
1	0100	Skin and appendages disorders	1174	1-2	899	787.4	2.23	0.009
2	0200	Musculo-skeletal system disorders	26	18-192	14	1.7	18.05	0.001
3	0200_0063	Arthralcia	15	18-192	11	0.98	41.9	0.001
4	0600	Gastro-intestinal system disorders	646	2-26	373	255.96	2.68	0.001
5	0600_0205	Diarrhoea	138	3-20	92	36.31	6.3	0.001
6	0700	Liver and biliary system disorders	41	4-189	29	10.54	7.25	0.009
7	0800	Metabolic and nutritional disorders	14	74-367	8	0.34	60.21	0.001
8	1220	White cell and RES disorders	38	4-313	32	9.83	15.99	0.001
9	1220_0572	Granulocytopenia	15	13-313	10	1.4	20.19	0.007
10	1220_0576	Leukocytosis	2	54-67	2	0.0091	infinity	0.043

Obs, Observed count; Exp, Expected count; RR, Relative Risk

6. Conclusion and Discussion

With PMS becoming more important, studies using the tree-based scan statistics are actively carried out. This method is particularly effective in detecting unusual or rare AEs in large healthcare data.

Many researchers use TreeScan software to implement this statistic. GUI-type environments have the advantage of being easy to use, but they often restricted from using programs. Users have to pre-process input data with another program. It is cumbersome and time-consuming to analyze many datasets. In case users have numerous datasets to analyze or want to apply multiple models, setting parameters and entering data per analysis can be quite tedious. To address these inconveniences, we developed a new and convenient R package called "rtreescan".

The package allows users to perform all analysis processes, including data pre-processing, analysis, and visualization, in one program. It saves researchers time and provides convenience in the analysis. It also enables analysis of many datasets and simulation studies are possible with repetitive tasks. This will contribute to the development of the tree-based scan statistic. When using data with large tree structures, such as real data, it is impossible to visualize the results. In this case, the function to visualize only the detected signals in the package can be useful. The figure will give users to aid in the understanding and interpretation of the result.

For the first time in this study, we also outlined mathematical formulas for all methods of this statistic. The tree-based scan statistic has been recently developed, so it is not easy to find related papers. Therefore, for all methods, this paper briefly describes hypotheses and characteristics and summarizes formulas.

In conclusion, this study is expected to help provide an overall understanding of the tree-based scan statistic. And the developed package will useful tool to conduct a study using this statistic. We hope this package will provide a convenient research environment to facilitate future studies by researchers.

7. Supplementary

Section 7 contains the full content of the vignette. This document is a helpful tutorial for users who are not familiar with analysis procedures and how to use the package. It is built into the package and can be modified later because it is a pilot version.

rtreescan

TreeScan™ is a data mining software that allows users to analyze large datasets using a different version of the tree-based scan statistic. This method scans both all individual cells and groups of closely related cells to find excess risk. It is mainly used to detect potential adverse events after drugs and vaccines are marketed. However, this can be applied to any data if the independent variable is a hierarchical structure. For more details, please refer to the paper on tree-based scan statistic.

rtreescan allows TreeScan™ to operate in an R environment, making it convenient and flexible. The functions in the package can be grouped into three main parts: parameter functions that set parameters for TreeScan™ or write them in a file to the OS; execution functions that run TreeScan™ in OS and generate output files; plotting functions that visualizes the structure of a tree.

rtreescan only does anything useful if you have TreeScan™. It is not open-source, but it is distributed free of charge. You can be downloaded for free from the link. <https://www.treescan.org>

```
library("rtreescan")
## rtreescan only does anything useful if you have TreeScan
## See http://www.treescan.org/ for free access
```

Basic usage of the package will:

1. use the ts.options() function to set TreeScan™ parameters; these are saved in R
2. use the write.ts.prm() function to write the TreeScan™ parameter file
3. use the treescan() function to run TreeScan™
4. use summary() on the treescan() object and proceed to analyze the results from TreeScan™ in R

Example data

Example data sets are automatically downloaded to your computer together with TreeScan™ itself. The input file format is shown below.

1. Tree Scan Statistic, Poisson Model

Count file Format : <nodeID>, <#cases>, <#population>

Tree file Format : <nodeID>, <parentNodeID>

The count and population data for each leaf node as shown below *(Count file: Poisson.cas)*.

```
head(Poisson)
##   nodeID cases population

## 1  Leaf1     1        200

## 2  Leaf2     2        300

## 3  Leaf3     0        400

## 4  Leaf4     6        200

## 5  Leaf5     4        300

## 6  Leaf6     2        200
```

The parameter file defines the probability model, input data, and output options to be performed in the TreeScan™.

If you have a prm file, you can simply use the treescan function as follows:

```
treescan(loc, "Poisson")
```

In the treescan(prmlocation, prmfilename), enter the location of the parameter file and the the name of file excluding the extension.

```
result <- treescan(loc, "Poisson")
result$csv
##   Cut.No. Node.Identifier Tree.Level Observed.Cases Expected Relative.Risk
## 1       1           Leaf9          4             14     2.68          6.66
## 2       2           Node3          3             33    16.10          3.63
## 3       3           Node6          2             35    25.49          2.03
## 4       4          Leaf10          4              7     2.68          2.84
## 5       5           Leaf4          4              6     2.68          2.39
## 6       6           Leaf8          4              9     5.37          1.81
##   Excess.Cases Log.Likelihood.Ratio P.value Ancestry.Order
## 1        11.90            13.136308   0.001              6
## 2        23.90            11.148394   0.001              3
## 3        17.73             3.318753   0.084              2
## 4         4.54             2.579169   0.191              4
## 5         3.49             1.619486   0.506              1
## 6         4.03             1.156791   0.633              5
```

If the option “results-csv”=‘y’ is set in the prm file, the result can also be seen in R.

If you don’t have a prm file, you can simply use the TBSS function as follows:

```
result <- TBSS(Poisson, tree, "Poisson", model=0, condition=1)
```

Arguments within the TBSS function are entered in order of case file, tree file, and file name of the parameter you want to create, model type, and condition type.

Also, users can also create functions in the following form by setting only the options they want:

The probability model and various options can be set based on the information in ts.options().

```
head(ts.options())
## [1] "[Analysis]"
## [2] ";scan type (TREEONLY=0, TREETIME=1, TIMEONLY=2)"
## [3] "scan-type=0"

## [4] ";probability model type (POISSON=0, BERNOULLI=1, UNIFORM=2,

        Not-Applicable=3)"

## [5] "probability-model=0"

## [6] ";conditional type (UNCONDITIONAL=0, TOTALCASES=1, NODE=2,

        NODEANDTIME=3)"

## [7] "conditional-type=1”
```

For example, if you want to set up a model and result options like this:

- model: conditional(“conditional-type”=1) Poisson model(“probability-model”= 0)
- result to save: csv and html file

you can set the options as the imp_treescan() function below.

```
imp_treescan <- function(cas, tre, filename){

  write.cas(cas, loc, filename)
  write.tre(tre, loc, filename)

  ts.options(list("tree-filename" = paste0(filename, ".tre"),
  "count-filename" =  paste0(filename, ".cas")))
  ts.options(list("probability-model"= 0, "conditional-type"= 1))
  ts.options(list("apply-risk-window-restriction"='y'))
  ts.options(list("results-csv"='y', "results-html"='y'))

  write.ts.prm(loc, filename)
  result <- treescan(loc, filename)
  return(result)
}

imp_treescan(Poisson, tree, "Poisson")
```

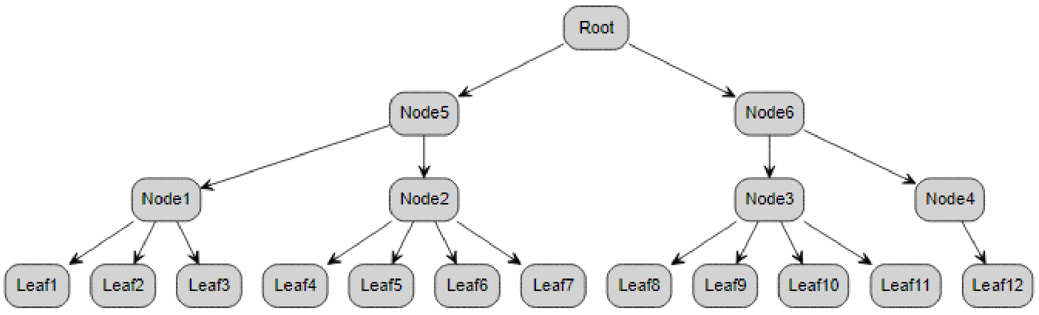
In the imp_treescan(count, tree, filename), enter the name of count file, tree file and prm file. Then the prm file is created with the option you set.

```
##                          _____________________________
##                                 TreeScan v1.4
##                          _____________________________
##
## Tree Only Scan with Conditional Poisson Model
##
________________________________________________________________________________
## SUMMARY OF DATA
##
## Total Cases...................: 55
## Total Expected................: 55.0
## Number of Nodes...............: 19
## Number of Root Nodes..........: 1
## Number of Nodes with Children.: 7
## Number of Leaf Nodes..........: 12
## Number of Levels in Tree......: 4
## Nodes per Levels..............: 1, 2, 4, 12
________________________________________________________________________________
## MOST LIKELY CUTS
##
## 1)Node Identifier.......: Leaf9
##   Tree Level............: 4
##   Observed Cases........: 14
##   Expected..............: 2.68
##   Relative Risk.........: 6.66
##   Excess Cases..........: 11.90
##   Log Likelihood Ratio..: 13.136308
##   P-value...............: 0.001
## ...
_________________________________________________________________________
```

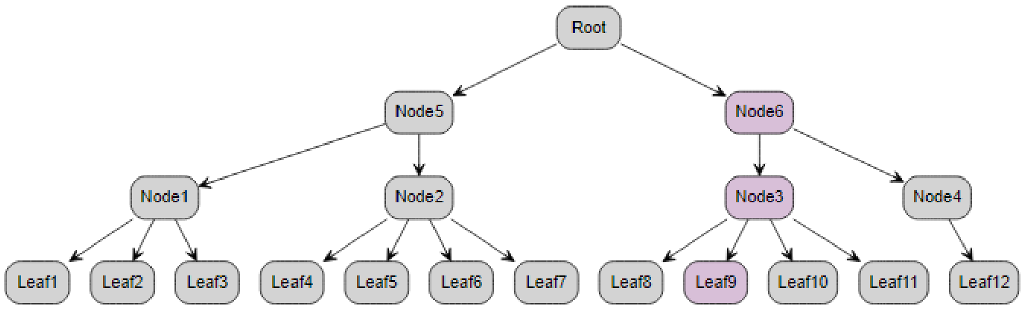
```
## PARAMETER SETTINGS
##
## Analysis
## --------
##   Type of Scan             : Tree Only
##   Conditional Analysis     : Total Cases
##   Probability Model - Tree : Poisson
## ...
```

To plot the treescan results, create the full tree structure in advance using maketree(). Then you can also use the makeSignaltree() to create a tree structure in which the signals are captured.

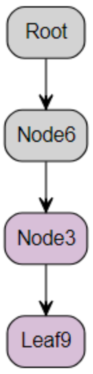
```
result <- treescan(loc, "Poisson")
entiretree <- maketree(tree)
plot(entiretree)
```



```r
signaltree <- makeSignaltree(result, entiretree, pv = 0.1, onlysignal = FALSE)
plot(signaltree)
```



```r
onlysignaltree <- makeSignaltree(result, entiretree, onlysignal = TRUE)
plot(onlysignaltree)
```



If the onlysignal option is set to **FALSE**, it shows the entire tree where the signal is captured, and if set to **TRUE** only shows where the signal is captured (default = FALSE). You can also use the desired p-value level using the pv option. In this example, we changed the p-value level to 0.1.

2.Tree Scan Statistic, Bernoulli Model

Count file Format : <nodeID>, <#cases>, <#controls>

Tree file Format : <nodeID>, <parentNodeID>

The case and control count data for each leaf node as shown below *(Count file: Bernoulli.cas)*.

```
head(Bernoulli)
##   nodeID cases controls
## 1  Leaf1     1        2
## 2  Leaf2     2        3
## 3  Leaf3     0        4
## 4  Leaf4     6        2
## 5  Leaf5     4        3
## 6  Leaf6     2        2
```

There is a '\t' in front of the parent node name in the example tree file, which requires a text cleaning process to remove it. The tree.tre file is also used in other model *(tree file: tree.tre)*.

```
head(tree)
##   nodeID parentNodeID
## 1  Leaf1        Node1
## 2  Leaf2        Node1
## 3  Leaf3        Node1
## 4  Leaf4        Node2
## 5  Leaf5        Node2
## 6  Leaf6        Node2
```

If you have a prm file, you can simply use the treescan function as follows:

```
treescan(loc, "Bernoulli")
```

In the treescan(prmlocation, prmfilename), enter the location of the parameter file and the the name of file excluding the extension.

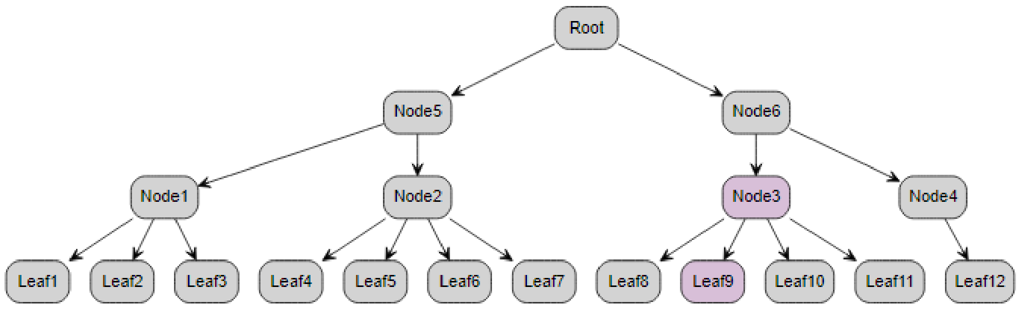
If you don't have a prm file, you can simply use the TBSS function as follows:

```
result <- TBSS(Bernoulli, tree, "Bernoulli", model=1, condition=0, self='y')
```

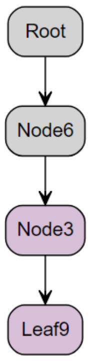
Arguments within the TBSS function are entered in order of case file, tree file, and file name of the parameter you want to create, model type, condition type, and self-control design status.

To plot the treescan results, create the full tree structure in advance using maketree(). Then You can also use the makeSignaltree() to create a tree structure in which the signals are captured.

```
result <- treescan(loc, "Bernoulli")
entiretree <- maketree(tree)
signaltree <- makeSignaltree(result, entiretree, pv = 0.1, onlysignal = FALSE)
plot(signaltree)
```



```
onlysignaltree <- makeSignaltree(result, entiretree, onlysignal = TRUE)
plot(onlysignaltree)
```



If the onlysignal option is set to **FALSE**, it shows the entire tree where the signal is captured, and if set to **TRUE** only shows where the signal is captured (default = FALSE).

3.Tree-Temporal Scan Statisitic

Count file Format : <nodeID>, <#cases>, <#time of cases>

Tree file Format : <nodeID>, <parentNodeID>

Study period: 1 to 28

The case and time of cases data for each leaf node as shown below *(Count file: TreeTemporal.cas)*.

```
head(TreeTemporal)
##   nodeID cases timeOfcases

## 1  Leaf1     1           6

## 2  Leaf1     1          16

## 3  Leaf1     1          23

## 4  Leaf2     2           7

## 5  Leaf2     1          24

## 6  Leaf2     1          25
```

If you have a prm file, you can simply use the treescan function as follows:

```
treescan(loc, "TreeTemporal")
```

If you don't have a prm file, you can simply use the TBSS function as follows:

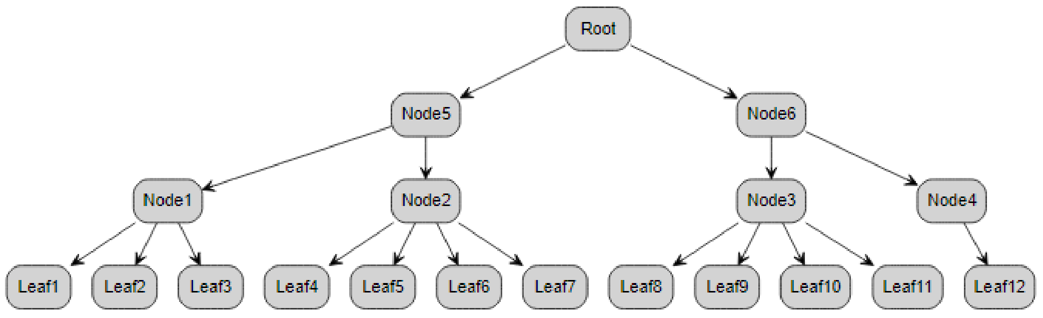
```
result <- TBSS(TreeTemporal, tree, "TreeTemporal", scan=1, model=2, condition=3,
start="[1,14]", end="[1,21]", entire="[1,28]")
```

Arguments within the TBSS function are entered in order of case file, tree file, and file name of the parameter you want to create, scan type, model type, condition type, and start range, end range, total period.

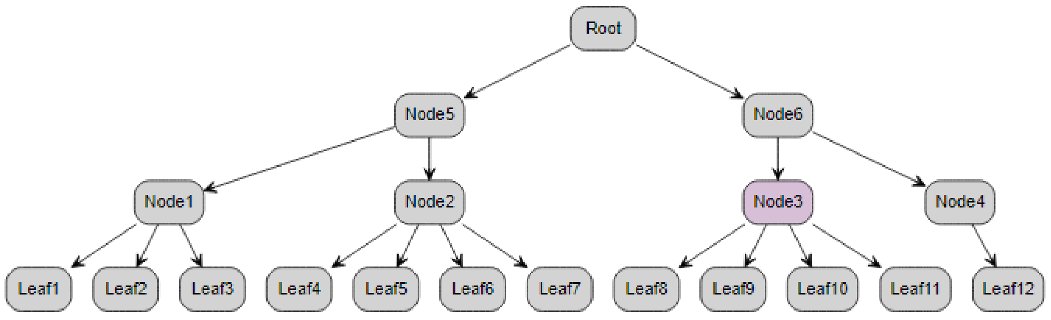
To plot the treescan results, create the full tree structure in advance using maketree(). Then You can also use the makeSignaltree() to create a tree structure in which the signals are captured.

```
result <- treescan(loc, "TreeTemporal")
entiretree <- maketree(tree)
```

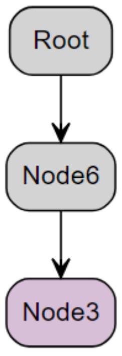
```
plot(entiretree)
```



```
signaltree <- makeSignaltree(result, entiretree, onlysignal = FALSE)
plot(signaltree)
```



```
onlysignaltree <- makeSignaltree(result, entiretree, onlysignal = TRUE)
plot(onlysignaltree)
```



If the onlysignal option is set to **FALSE**, it shows the entire tree where the signal is captured, and if set to **TRUE** only shows where the signal is captured (default = FALSE).

Simulated data: Pharmacovigilance data set

The Korea Adverse Events Reporting System (KAERS) is designed that can report and manage information about unusual cases after taking medicine and medical supplies. The KAERS was coded based on **WHO Adverse Reactions Terminology (WHO-ART)** for internationally standardized classification. WHO-ART is a dictionary meant to serve as a basis for rational coding of adverse reaction terms. It has a 4 level hierarchical structure that begins with the organ system of human.

- SOC: System-organ classesbody organ groups
- HT: High level terms for grouping preferred terms
- PT: Preferred terms principal terms for describing adverse reactions
- IT: Included terms synonyms to Preferred terms

We considered only 3 level tree structure considering PT, HT, and SOC terms, and extracted only a few trees with 14 leaves from the full tree. It also was simulated with different values, while maintaining the pattern of the actual number of spontaneous reports for each leaf.

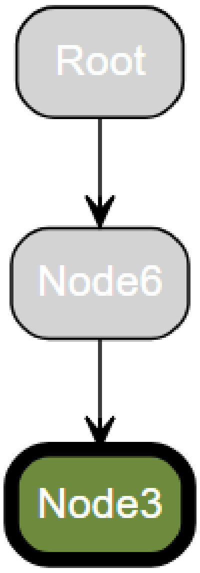
```
data(PVcas, package = "rtreescan")
data(Pvtre, package = "rtreescan")
head(Pvcas)
##          nodeID cases controls
## 1 0600_0268_003   153       34
## 2 1810_2237_002    96       26
## 3 1100_1976_001     5       16
## 4 1100_2328_001     3        6
## 5 1810_1058_001     5       10
## 6 1820_0054_006     1        8
head(Pvtre)
##          nodeID parentNodeID
## 1 1820_0054_001    1820_0054
## 2 1820_0054_002    1820_0054
## 3 1820_0054_003    1820_0054
## 4 0410_0115_001    0410_0115
## 5 0600_0268_003    0600_0268
## 6 0600_1014_005    0600_1014
```

```
result <- TBSS(PVcas, PVtre, "Bernoulli", model=1, condition=0, self='y')
result$csv
##   Cut.No. Node.Identifier Tree.Level Observations Cases Expected
## 1       1   0600_0268_003          4          187   153     93.5
## 2       2       0600_0268          3          187   153     93.5
## 3       3            0600          2          187   153     93.5
## 4       4   1810_2237_002          4          122    96     61.0
## 5       5       1810_2237          3          122    96     61.0
## 6       6            Root          1          491   317    245.5
## 7       7            1810          2          165   110     82.5
## 8       8   1820_0054_001          4           14    13      7.0
##   Relative.Risk Excess.Cases Log.Likelihood.Ratio P.value Ancestry.Order
## 1          4.50          119            40.954471   0.001              4
## 2          4.50          119            40.954471   0.001              3
## 3          4.50          119            40.954471   0.001              2
## 4          3.69           70            21.361325   0.001              7
## 5          3.69           70            21.361325   0.001              6
## 6          1.82          143            21.128683   0.001              1
## 7          2.00           55             9.344447   0.001              5
## 8         13.00           12             6.101600   0.011              8

entiretree <- maketree(PVtre)
signaltree <- makeSignaltree(result, entiretree, onlysignal=F)
```

Plot style setting

You can change the style of a node or edge on the tree.

```
signaltree <- makeSignaltree(result, entiretree, onlysignal=F, fillcolor=
fillcolor = "darkolivegreen4")


SetNodeStyle(onlysignaltree$Node6$Node3, fillcolor = "darkolivegreen4",
penwidth = "5px")
plot(onlysignaltree)
```



In the makeSignaltree(), you can change the color of the node using the fillcolor option, and the thickness of the edge with the penwidth option, as shown above. It is also possible to change color only certain nodes can change color. There are many options for visualizing the tree structure.

Please refer to the following link for more options.

https://cran.r-project.org/web/packages/data.tree/vignettes/data.tree.html

References

Abrams, A. M., Kleinman, K., & Kulldorff, M. (2010). Gumbel based p-value approximations for spatial scan statistics. *International Journal of Health Geographics*, 9(1), 61.

Alomar, M., Tawfiq, A. M., Hassan, N., & Palaian, S. (2020). Post marketing surveillance of suspected adverse drug reactions through spontaneous reporting: current status, challenges and the future. *Therapeutic Advances in Drug Safety*, 11, 2042098620938595.

Arnaud, M., Bégaud, B., Thurin, N., Moore, N., Pariente, A., & Salvo, F. (2017). Methods for safety signal detection in healthcare databases: a literature review. *Expert opinion on drug safety*, 16(6), 721-732.

Brown, J. S., Petronis, K. R., Bate, A., Zhang, F., Dashevsky, I., Kulldorff, M., ... & Boudreau, D. (2013). Drug adverse event detection in health plan data using the Gamma Poisson Shrinker and comparison to the tree-based scan statistic. *Pharmaceutics*, 5(1), 179-200.

Ding, Y., Markatou, M., & Ball, R. (2020). An evaluation of statistical approaches to postmarketing surveillance. *Statistics in Medicine*, 39(7), 845-874.

Dwass, M. (1957). Modified randomization tests for nonparametric hypotheses. *The Annals of Mathematical Statistics*, 181-187.

Huang, L., Guo, T., Zalkikar, J. N., & Tiwari, R. C. (2014). A review of statistical methods for safety surveillance. *Therapeutic innovation & regulatory science*, 48(1), 98-108.

Jung, I. (2019). Spatial scan statistics for matched case-control data. PloS one, 14(8), e0221225.

Kulldorff, M. (1997). A spatial scan statistic. Communications in Statistics-Theory and methods, 26(6), 1481-1496.

Kulldorff, M., Fang, Z., & Walsh, S. J. (2003). A tree-based scan statistic for database disease surveillance. *Biometrics*, 59(2), 323-331.

Kulldorff, M., Dashevsky, I., Avery, T. R., Chan, A. K., Davis, R. L., Graham, D., ... & Herrinton, L. J. (2013). Drug safety data mining with a tree-based scan statistic. *Pharmacoepidemiology and drug safety*, 22(5), 517-523.

Kulldorff M. and Information Management Services, Inc. TreeScanTM v1.4: Software for the tree based scan statistic. http://www.treescan.org/, 2018.

Lee, H., Kim, J. H., Choe, Y. J., & Shin, J. Y. (2020). Safety Surveillance of Pneumococcal Vaccine Using Three Algorithms: Disproportionality Methods, Empirical Bayes Geometric Mean, and Tree-Based Scan Statistic. Vaccines, 8(2), 242.

Li, Y., Ryan, P. B., Wei, Y., & Friedman, C. (2015). A method to combine signals from spontaneous reporting systems and observational healthcare data to detect adverse drug reactions. *Drug safety*, 38(10), 895-908.

Mahaux, O., Bauchau, V., Zeinoun, Z., & Van Holle, L. (2019). Tree-based scan statistic–Application in manufacturing-related safety signal detection. *Vaccine*, 37(1), 49-55.

Maro, J. C., Nguyen, M. D., Dashevsky, I., Baker, M. A., & Kulldorff, M. (2017). Statistical power for postlicensure medical product safety data mining. *eGEMs*, *5*(1).

Park, G. (2019). A tree-based scan statistic for zero-inflated count data in post-market drug safety surveillance. Ph.D.diss., Yonsei University, Seoul, Republic of Korea.

Park, G., Jung, H., Heo, S. J., & Jung, I. (2020). Comparison of Data Mining Methods for the Signal Detection of Adverse Drug Events with a Hierarchical Structure in Postmarketing Surveillance. *Life*, *10*(8), 138.

Prates, M. O., Assunção, R. M., & Costa, M. A. (2012). Flexible scan statistic test to detect disease clusters in hierarchical trees. *Computational Statistics*, 27(4), 715-737.

Polprasert, C., & Prayongratana, K. (2009). Levofloxacin-induced severe thrombocytopenia. *J Med Assoc Thai*, 92(suppl 3), S69-71.

Roux, E., Thiessard, F., Fourrier, A., Bégaud, B., & Tubert-Bitter, P. (2007). Spontaneous Reporting System Modelling for the Evaluation of Automatic Signal Generation Methods in Pharmacovigilance. *Advances in Statistical Methods for the Health Sciences,* 75-92*,* Birkhäuser Boston.

Sakaeda, T., Tamon, A., Kadoyama, K., & Okuno, Y. (2013). Data mining of the public version of the FDA Adverse Event Reporting System. International journal of medical

국 문 요 약

TreeScan software를 R에서 구현하는 패키지의 개발과 적용

Kulldorff (2003)가 제안한 트리 기반 검색 통계량은 계층적 형태의 데이터에서 과도한 위험 신호를 탐지하는 유용한 데이터 마이닝 방법이다. 이 방법은 시판 후 의약품 안전성 감시에서 전통적인 방법보다 빠르게 의약품 안전 문제를 감지할 수 있는 "조기 경고 시스템 (early warning system)"으로서 중요한 역할을 한다. 이러한 목적으로 국제 표준 의약품 부작용 분류 체계 (WHO-ART, MedDRA)로 보고된 이상 반응 자료는 효과적이다.

TreeScan은 다양한 버전의 트리 기반 검색 통계량을 구현하는 소프트웨어이다. 하지만 그래픽 사용자 인터페이스 (GUI) 설계로 인해 사용상 불편함과 제한점이 있다. TreeScan에서는 데이터 전처리가 불가하므로 다른 프로그램을 사용하여 입력 데이터를 정해진 포맷으로 변환해야 한다. 또한, 시뮬레이션 연구와 같이, 여러 데이터셋을 사용하여 반복적인 분석을 수행하기 어렵다는 문제점이 있다. 우리는 앞서 언급한 여러 불편 요소를 해결하는 "rtreescan" R 패키지를 소개한다.

본 연구는 두 가지 주요 목적을 가진다. 첫 번째는 R 프로그램에서 TreeScan이 구현되는 "rtreescan" 패키지를 개발하는 것이다. 두 번째는 각 모형의 특징을 알아보고 모형별 통계량 및 상대 위험 (Relative Risk)과 같은 수학 공식을

개략적으로 서술하는 것이다. 그리고 2010년부터 2016년까지 한국의약품안전관리원에 보고된 의약품부작용보고원시자료 (KIDS-KD)에 대해 각 방법을 적용하여 분석 결과를 설명한다.

결론적으로, 우리는 데이터 전처리와 분석 그리고 시각화를 포함한 모든 분석 프로세스를 하나의 프로그램에서 수행할 수 있는 패키지를 개발하였다. 시뮬레이션 연구가 가능해지면서 트리 기반 검색 통계량의 발전에 기여할 수 있을 것으로 보인다. 또한, 이 연구에서 처음으로 트리 기반 검색 통계량의 모든 방법에 대한 공식을 요약하였다. 본 연구가 트리 기반 검색 통계량에 대한 전반적인 이해와 향후 연구자들의 수월한 연구 진행에 도움이 될 것으로 기대한다.

핵심되는 말 : 트리 기반 검색 통계량, 이상 사례, 시판 후 의약품 안전성 감시, 약물 감시, 신호 탐지