tsdataleaks: An R Package to Detect Potential Data Leaks in Forecasting Competitions

5 February 2024

Summary

Forecasting competitions are of increasing importance as a means to learn best practices and gain knowledge. Data leakage is one of the most common issues that can often be found in competitions. Data leaks can happen when the training data contains information about the test data. There are a variety of different ways that data leaks can occur with time series data. For example: i) randomly chosen blocks of time series are concatenated to form a new time series; ii) scale-shifts; iii) repeating patterns in time series; iv) white noise is added to the original time series to form a new time series, etc. This work introduces a novel tool to detect these data leaks. The tsdataleaks package provides a simple and computationally efficient algorithm to exploit data leaks in time series data. This paper demonstrates the package design and its power to detect data leakages with an application to forecasting competition data.

Statement of Need

Time series forecasting competitions have played a significant role in the advancement of forecasting practices. Typically, in forecasting competitions, a collection of time series is given to the participants, and then the participants submit the forecasts for the required test period for each time series. During the competition period, only the training set for each time series is given to the public, and the test set is kept private from the public. Finally, competition organizers evaluate the forecast accuracy by comparing each participants submitted forecast values against the actual test period values. Participating in forecasting competitions not only aids in the identification of novel methods and facilitates their performance comparison against existing state-of-the-art forecasting techniques, as highlighted by R. J. Hyndman (2020), but also provides empirical evidence crucial for enhancing forecasting performance and advancing the theory and practice of forecasting (Makridakis, Spiliotis, and Assimakopoulos 2022).

Data leakage occurs when the training period of the time series includes test period data before officially releasing the test period of the time series. This idea is illustrated in Figure 1. A and B are two time series. The latter segment of the training set and the subsequent test set within the (A) series are the same as the red segment highlighted in the training segment inherent to series (B). This type of data leak could occur when randomly chosen blocks of time series are concatenated to form a new time series.

Competitions with data leaks will not be able to reach their original purpose. By exploiting data leakage, competitors can obtain a top rank in the leaderboard. Such models look highly accurate within the competition environment but become inaccurate when applied to a data set outside the competition environment. Hence, there is an increasing need to examine the potential data leaks in time series before the release of data to the public. The tsdataleaks package is designed to identify data leaks in time series.

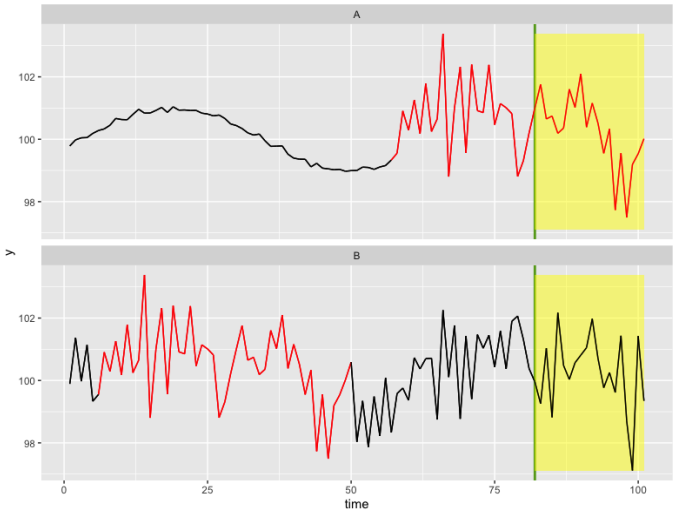

Figure 1: An example of a time series data leak. "A" and "B"" are two time series. The green vertical line and yellow background separate the training and test parts of the series. A training segment of series B (red colour segment) is the source of the latter segment of the training set and test set of the A series.

State of the Field in R

As of the latest information available on the Comprehensive R Archive Network (CRAN) Task View: Time Series Analysis (Rob J Hyndman 2023), there is no package available for detecting data leakages.

Algorithm

The algorithm operates as follows: the algorithm begins by selecting the final segment of the training section in each time series within the collection. Subsequently, those segments iterates through all of the time series one time step at a time and calculate Pearson's correlation coefficient. Hence, the inputs to the algorithm are: i) the time series collection; ii) segment length; and iii) the cut-off value for the correlation coefficient. A data leak is indicated if the Pearson's correlation coefficient's absolute value exceeds the cutoff value. The algorithm returns the starting and end indexes of the segments that match each time series' last segment corresponds to the training part.

Algorithm: Time Series Matching

Input:
1. *lstx*: A collection of time series as a list in R.
2. *h*: Length of the segment to be considered.
3. *cutoff*: Cut-off value for the absolute value of the Pearson's correlation coefficient.

Output:
A list containing starting and ending indices of segments that match each time series' last segment of the training set.

Steps:
1. Extract the last segment of the training part with length *h*.
2. *Loop through the time series* with one time step, considering each segment:
- Calculate the Pearson's correlation coefficient between the current segment of a time series and the last portion of the training segment that was retrieved.

Algorithm: Time Series Matching

- If the correlation coefficient is above the *cutoff*:
- Return the matching segments list with the starting and ending indices of the matching segments.

3. *Return* the matching segments list as the output.

Figure 2 illustrates the first iteration of the algorithm. The correlation between the purple segment and observations 1–6 (dark green section) of the first time series is measured at the first iteration.

Figure 3 visualize the second iteration of the algorithm. The correlation between the purple segment and observations 2–7 (dark green section) of the first time series is measured at the second iteration. Figure 4 illustrates an intermediate step of the algorithm.

Figure 2: Visualization of the first iteration of the algorithm. The last segment of the training part of the first series is colored purple. As the first step of the algorithm, it computes Pearson's correlation coefficient between the observations 1-6 (dark green section) and the purple segment.

Figure 3: Visualization of the second iteration of the algorithm. The last segment of the training part of the first series is colored purple. As the second step of the algorithm, it computes Pearson's correlation coefficient between the observations 2-7 (dark green section) and the purple segment.

Figure 4: Intermediate step of the algorithm: identification of potential data leaks. The light purple section of the fourth series perfectly correlates with the last segment of the first series. Hence, the red section of the fourth series could be the test part of the first series.

Usage

Installation

The package tsdataleaks is available on both CRAN and GitHub and can be installed and loaded into the R session using:

```
install.packages("tsdataleaks")
library(tsdataleaks)
```

or

```
devtools::install_github("thiyangt/tsdataleaks")
library(tsdataleaks)
```

Functionality

There are three functions in the package: i) `find_dataleaks`, ii) `viz_dataleaks`, and iii) `reason_dataleaks`. To demonstrate the package functions, a small time series collection with three time series is created.

```
set.seed(2024)
x <- rnorm(15)
lst <- list(
  x = x,
  y = c(rnorm(10), x[1:5]),
  z = c(rnorm(10), x[10:15]))
```

Following are the steps for detecting data leakages and visualizing the results.

Step 1: The main function in the package is `find_dataleaks`. It exploits the data leakages according to the algorithm. The inputs to the function are a list of time series collections (lstx), the length of the segment to be considered (h), and the cutoff value for the absolute value of Pearson's correlation coefficient (cutoff). The `f1` output is shown in Figure 5 (step1).

```
f1 <- find_dataleaks(lstx = lst, h=5, cutoff=1)
```

Step 2: `viz_dataleaks` function arranges the results of `find_dataleaks` in matrix form and visualizes them for easy understanding, as shown in Figure 5 (step2).

```
viz_dataleaks(f1)
```

Step 3: `reason\_dataleaks` displays the reasons for data leaks and evaluates the usefulness of data leaks towards the winning of the competition. The inputs to the function are a list of time series collections (lstx), length of the segment to be considered (h), output of the find_dataleaks function (finddataleaksout). The corresponding outputs are shown in Figure 5 (step3).

```
reason_dataleaks(lstx = lst, finddataleaksout = f1, h=5)
```

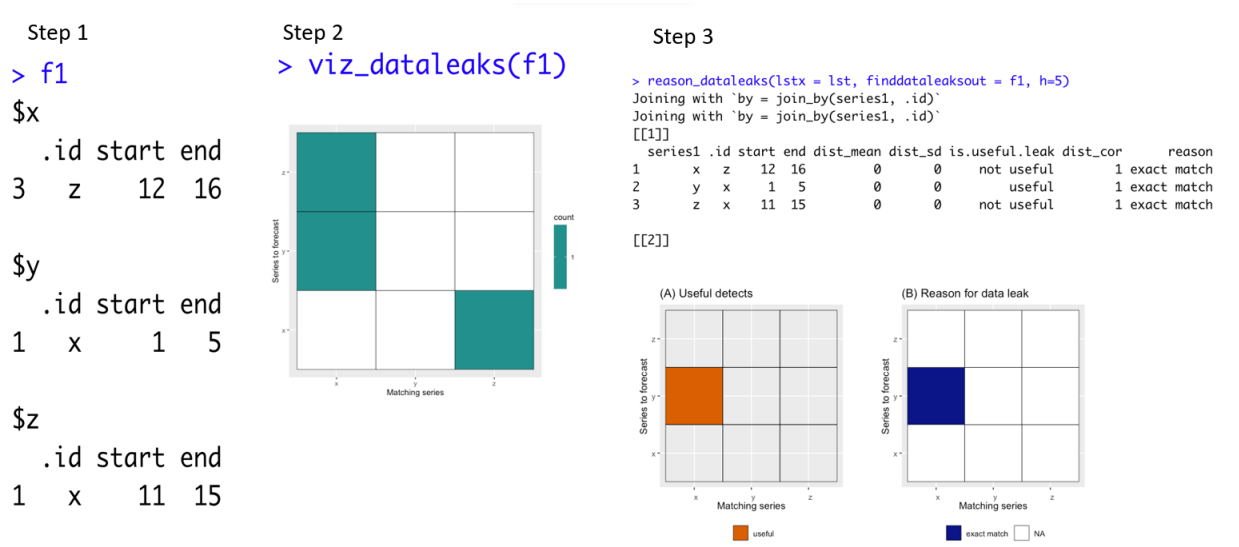

Figure 5: The outputs of f1, viz_dataleaks(f1) and reason_dataleaks(lstx = lst, finddataleaksout = f1, h=5)

According to the Figure 5 step 1 results, the last part of the training set in x series perfectly correlates with z series: 12–16 observations; the last part of y series correlates with x series: 1–5 observations; and the last part of z series correlates with x series: 11–15 observations. However, according to the step 3 results, only "the last part of the y series correlates with the x series: 1–5 observations" identification is useful in winning the competition. The reason is that we have x-series 6:10 observations. Hence, we can use that as the test value of the y series. "The last part of the z series correlates with x series: 11–15 observations"-This identification is not useful in winning the competition because x series: 16–20 observations are not available. In this example, all data leakages occur due to an exact match.

Appication to the M1 Competition Yearly Time Series Data

M-competitions is a series of time-series forecasting competitions organized by Spyros Makridakis and his team (Makridakis, Spiliotis, and Assimakopoulos 2020). M1-competition data is available in the package Mcomp (R. Hyndman 2018). Before applying the find_dataleaks function, all of the training sets of yearly series are stored in a list called `M1Y_x`. In the M1 competition, the length of the test period for the yearly

series is 6. Hence, the `h` value is selected as 6. The cutoff value for the absolute value of Pearson's correlation coefficient is 1. The results are shown in Figure 6.

```
library(Mcomp)
data("M1")
M1Y <- subset(M1, "yearly")
M1Y_x <- lapply(M1Y, function(temp){temp$x})
m1y_f1 <- find_dataleaks(M1Y_x, h=6, cutoff = 1)
viz_dataleaks(m1y_f1)
reason_dataleaks(M1Y_x, m1y_f1, h=6, ang=90)
```

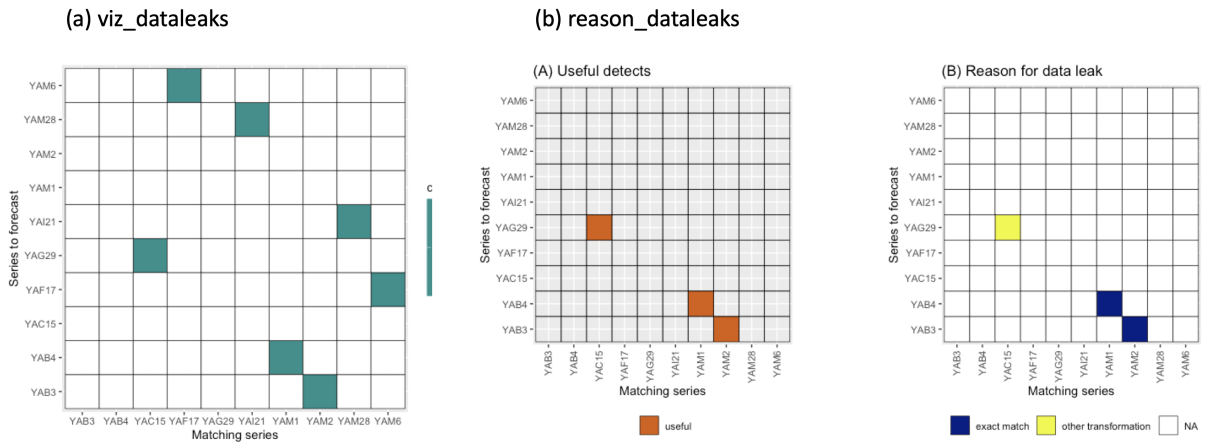

Figure 6: Outputs of viz_dataleaks(m1y_f1) and reason_dataleaks(M1Y_x, m1y_f1, h=6, ang=90).

According to Figure 6, there are 7 data leakage detentions. Out of them, only three are useful in winning the competition. Two data leakages are due to an exact match; the other is due to a linear transformation of the form $y = mx + c$.

Documentation and Examples

Applications to other examples can be found in the README.md file at https://github.com/thiyangt/tsdataleaks.

Conclusion

The new open-source R package described in this paper enables: i) exploiting data leakages; ii) identifying the reasons for data leakage as exact matches or adding a constant or other transformations. iii) determining whether the data leakages identified are useful in winning the forecast competition; and iv) visualizing the results. The R package tsdataleaks is a valuable tool for competition organizers to avoid data leakages, participants to detect data leakages, and the entire forecasting research community to evaluate the quality of data.

Reproducibility

Codes to generate this manuscript is available at https://github.com/thiyangt/tsdataleaks/blob/master/paper/paper.md

References

Hyndman, Rob. 2018. "Mcomp: Data from the m-Competitions." https://CRAN.R-project.org/package=Mcomp.

Hyndman, Rob J. 2020. "A Brief History of Forecasting Competitions." *International Journal of Forecasting* 36 (1): 7–14.

Makridakis, Spyros, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2020. "The M4 Competition: 100,000 Time Series and 61 Forecasting Methods." *International Journal of Forecasting* 36 (1): 54–74.

———. 2022. "M5 Accuracy Competition: Results, Findings, and Conclusions." *International Journal of Forecasting* 38 (4): 1346–64.

Rob J Hyndman, Rebecca Killick. 2023. "CRAN Task View: Time Series Analysis." https://cran.r-project.org/web/views/TimeSeries.html.