Cleansing the dataset

Choonghyun Ryu

2020-01-23

Preface

If you created a dataset to create a classification model, you must perform cleansing of the data. After you create the dataset, you should do the following:

- **Cleansing the dataset**
  - **Optional removal of variables including missing values**
  - **Remove a variable with one unique number**
  - **Remove categorical variables with a large number of levels**
  - **Convert a character variable to a categorical variable**

- Split the data into a train set and a test set
- Modeling and Evaluate, Predict

The alookr package makes these steps fast and easy:

Data: create example dataset

To illustrate basic use of the alookr package, create the `data_exam` with sample function. The `data_exam` dataset include 5 variables.

variables are as follows.:

- `id` : character
- `year`: character
- `count`: numeric
- `alpha` : character
- `flag` : character

```
# create sample dataset
set.seed(123L)
id <- sapply(1:1000, function(x)
  paste(c(sample(letters, 5), x), collapse = ""))

year <- "2018"

set.seed(123L)
count <- sample(1:10, size = 1000, replace = TRUE)

set.seed(123L)
alpha <- sample(letters, size = 1000, replace = TRUE)

set.seed(123L)
flag <- sample(c("Y", "N"), size = 1000, prob = c(0.1, 0.9), replace = TRUE)

data_exam <- data.frame(id, year, count, alpha, flag, stringsAsFactors = FALSE)

# structure of dataset
str(data_exam)
'data.frame':   1000 obs. of  5 variables:
```

```
 $ id   : chr  "htjuw1" "bnvmk2" "ylqnc3" "xgbhu4" ...
 $ year : chr  "2018" "2018" "2018" "2018" ...
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
 $ alpha: chr  "h" "u" "k" "w" ...
 $ flag : chr  "N" "N" "N" "N" ...

# summary of dataset
summary(data_exam)
      id                year              count           alpha
 Length:1000        Length:1000        Min.   : 1.000   Length:1000
 Class :character   Class :character   1st Qu.: 3.000   Class :character
 Mode  :character   Mode  :character   Median : 5.000   Mode  :character
                                       Mean   : 5.474
                                       3rd Qu.: 8.000
                                       Max.   :10.000
    flag
 Length:1000
 Class :character
 Mode  :character
```

Clean dataset

`cleanse()` cleans up the dataset before fitting the classification model.

The function of cleanse() is as follows.:

- remove variables whose unique value is one
- remove variables with high unique rate
- converts character variables to factor
- remove variables with missing value

Cleanse dataset with `cleanse()`

For example, we can cleanse all variables in `data_exam`:

```
# cleansing dataset
newDat <- cleanse(data_exam)
── Checking unique value ──────────────────────── unique value is one ──
remove variables that unique value is one
• year

── Checking unique rate ──────────────────────────── high unique rate ──
remove variables with high unique rate
• id = 1000(1)

── Checking character variables ──────────────────── categorical data ──
converts character variables to factor
• alpha
• flag

# structure of cleansing dataset
str(newDat)
```

```
'data.frame':   1000 obs. of  3 variables:
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
 $ alpha: Factor w/ 26 levels "a","b","c","d",..: 8 21 11 23 25 2 14 24 15 12 ...
 $ flag : Factor w/ 2 levels "N","Y": 1 1 1 1 2 1 1 1 1 1 ...
```

- `remove variables whose unique value is one` : The year variable has only one value, "2018". Not needed when fitting the model. So it was removed.
- `remove variables with high unique rate` : If the number of levels of categorical data is very large, it is not suitable for classification model. In this case, it is highly likely to be an identifier of the data. So, remove the categorical (or character) variable with a high value of the unique rate defined as "number of levels / number of observations".
  - The unique rate of the id variable with the number of levels of 1000 is 1. This variable is the object of the removal by identifier.
  - The unique rate of the alpha variable is 0.026 and this variable is also removed.
- `converts character variables to factor` : The character type flag variable is converted to a factor type.

For example, we can not remove the categorical data that is removed by changing the threshold of the `unique rate`:

```
# cleansing dataset
newDat <- cleanse(data_exam, uniq_thres = 0.03)
── Checking unique value ──────────────────────── unique value is one ──
remove variables that unique value is one
• year

── Checking unique rate ──────────────────────── high unique rate ──
remove variables with high unique rate
• id = 1000(1)

── Checking character variables ──────────────────────── categorical data ──
converts character variables to factor
• alpha
• flag

# structure of cleansing dataset
str(newDat)
'data.frame':   1000 obs. of  3 variables:
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
 $ alpha: Factor w/ 26 levels "a","b","c","d",..: 8 21 11 23 25 2 14 24 15 12 ...
 $ flag : Factor w/ 2 levels "N","Y": 1 1 1 1 2 1 1 1 1 1 ...
```

The `alpha` variable was not removed.

If you do not want to apply a unique rate, you can set the value of the `uniq` argument to FALSE.:

```
# cleansing dataset
newDat <- cleanse(data_exam, uniq = FALSE)
── Checking character variables ──────────────────────── categorical data ──
converts character variables to factor
• id
• year
• alpha
• flag

# structure of cleansing dataset
```

```
str(newDat)
'data.frame':   1000 obs. of  5 variables:
 $ id   : Factor w/ 1000 levels "abety794","abkoe306",..: 301 59 929 890 904 694 997 465 134 124 ...
 $ year : Factor w/ 1 level "2018": 1 1 1 1 1 1 1 1 1 1 ...
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
 $ alpha: Factor w/ 26 levels "a","b","c","d",..: 8 21 11 23 25 2 14 24 15 12 ...
 $ flag : Factor w/ 2 levels "N","Y": 1 1 1 1 2 1 1 1 1 1 ...
```

If you do not want to force type conversion of a character variable to factor, you can set the value of the `char` argument to FALSE.:

```
# cleansing dataset
newDat <- cleanse(data_exam, char = FALSE)
── Checking unique value ──────────────────────── unique value is one ──
remove variables that unique value is one
• year

── Checking unique rate ─────────────────────────── high unique rate ──
remove variables with high unique rate
• id = 1000(1)

# structure of cleansing dataset
str(newDat)
'data.frame':   1000 obs. of  3 variables:
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
 $ alpha: chr  "h" "u" "k" "w" ...
 $ flag : chr  "N" "N" "N" "N" ...
```

If you want to remove a variable that contains missing values, specify the value of the `missing` argument as TRUE. The following example **removes the flag variable** that contains the missing value.

```
data_exam$flag[1] <- NA

# cleansing dataset
newDat <- cleanse(data_exam, missing = TRUE)
── Checking missing value ──────────────────────── included NA ──
remove variables whose included NA
• flag

── Checking unique value ──────────────────────── unique value is one ──
remove variables that unique value is one
• year

── Checking unique rate ─────────────────────────── high unique rate ──
remove variables with high unique rate
• id = 1000(1)

── Checking character variables ──────────────────── categorical data ──
converts character variables to factor
• alpha

# structure of cleansing dataset
str(newDat)
'data.frame':   1000 obs. of  2 variables:
 $ count: int  3 8 5 9 10 1 6 9 6 5 ...
```

```
 $ alpha: Factor w/ 26 levels "a","b","c","d",..: 8 21 11 23 25 2 14 24 15 12 ...
```

Diagnosis and removal of highly correlated variables

In the linear model, there is a multicollinearity if there is a strong correlation between independent variables. So it is better to remove one variable from a pair of variables where the correlation exists.

Even if it is not a linear model, removing one variable from a strongly correlated pair of variables can also reduce the overhead of the operation. It is also easy to interpret the model.

Cleanse dataset with `treatment_corr()`

`treatment_corr()` diagnose pairs of highly correlated variables or remove on of them.

`treatment_corr()` calculates correlation coefficient of pearson for numerical variable, and correlation coefficient of spearman for categorical variable.

For example, we can diagnosis and removal of highly correlated variables:

```
# numerical variable
x1 <- 1:100
set.seed(12L)
x2 <- sample(1:3, size = 100, replace = TRUE) * x1 + rnorm(1)
set.seed(1234L)
x3 <- sample(1:2, size = 100, replace = TRUE) * x1 + rnorm(1)

# categorical variable
x4 <- factor(rep(letters[1:20], time = 5))
set.seed(100L)
x5 <- factor(rep(letters[1:20 + sample(1:6, size = 20, replace = TRUE)], time = 5))
set.seed(200L)
x6 <- factor(rep(letters[1:20 + sample(1:3, size = 20, replace = TRUE)], time = 5))
set.seed(300L)
x7 <- factor(sample(letters[1:5], size = 100, replace = TRUE))

exam <- data.frame(x1, x2, x3, x4, x5, x6, x7)
str(exam)
'data.frame':   100 obs. of  7 variables:
 $ x1: int  1 2 3 4 5 6 7 8 9 10 ...
 $ x2: num  0.957 5.957 8.957 3.957 4.957 ...
 $ x3: num  -0.806 2.194 4.194 6.194 8.194 ...
 $ x4: Factor w/ 20 levels "a","b","c","d",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ x5: Factor w/ 17 levels "c","d","e","g",..: 1 2 4 3 5 6 8 7 9 8 ...
 $ x6: Factor w/ 14 levels "c","d","e","g",..: 1 2 3 4 5 6 7 6 8 8 ...
 $ x7: Factor w/ 5 levels "a","b","c","d",..: 5 4 5 4 4 1 4 3 3 5 ...
head(exam)
  x1        x2         x3 x4 x5 x6 x7
1  1 0.9573151 -0.8060313  a  c  c  e
2  2 5.9573151  2.1939687  b  d  d  d
3  3 8.9573151  4.1939687  c  g  e  e
4  4 3.9573151  6.1939687  d  e  g  d
5  5 4.9573151  8.1939687  e  h  h  d
6  6 5.9573151 10.1939687  f  i  i  a

# default case
exam_01 <- treatment_corr(exam)
```

```
* remove variables whose strong correlation (pearson >= 0.8)
 - remove x1 : with x3 (0.8072)
* remove variables whose strong correlation (spearman >= 0.8)
 - remove x4 : with x5 (0.9823)
 - remove x4 : with x6 (0.9955)
 - remove x5 : with x6 (0.9853)
head(exam_01)
         x2         x3 x6 x7
1 0.9573151 -0.8060313  c  e
2 5.9573151  2.1939687  d  d
3 8.9573151  4.1939687  e  e
4 3.9573151  6.1939687  g  d
5 4.9573151  8.1939687  h  d
6 5.9573151 10.1939687  i  a

# not removing variables
treatment_corr(exam, treat = FALSE)
* remove variables whose strong correlation (pearson >= 0.8)
 - remove x1 : with x3 (0.8072)
* remove variables whose strong correlation (spearman >= 0.8)
 - remove x4 : with x5 (0.9823)
 - remove x4 : with x6 (0.9955)
 - remove x5 : with x6 (0.9853)

# Set a threshold to detecting variables when correlation greater then 0.9
treatment_corr(exam, corr_thres = 0.9, treat = FALSE)
* remove variables whose strong correlation (spearman >= 0.9)
 - remove x4 : with x5 (0.9823)
 - remove x4 : with x6 (0.9955)
 - remove x5 : with x6 (0.9853)

# not verbose mode
exam_02 <- treatment_corr(exam, verbose = FALSE)
head(exam_02)
         x2         x3 x6 x7
1 0.9573151 -0.8060313  c  e
2 5.9573151  2.1939687  d  d
3 8.9573151  4.1939687  e  e
4 3.9573151  6.1939687  g  d
5 4.9573151  8.1939687  h  d
6 5.9573151 10.1939687  i  a
```

- `remove variables whose unique value is one` : The year variable has only one value, "2018". Not needed when fitting the model. So it was removed.