R 语言编程：基于 tidyverse

第 23 讲　数据清洗

张敬信

2022 年 3 月 11 日

哈尔滨商业大学

探索性数据分析 (Exploratory Data Analysis, EDA)

传统统计分析，是先假设样本服从某种分布，再将数据套入假设模型再做分析。但由于多数数据并不能满足假设的分布，因此结果常常不能让人满意。

而**探索性数据分析**[1]，更注重数据的真实分布，通过可视化、变换、建模来探索数据，发现数据中隐含的规律，从而得到启发找到适合数据的模型。

EDA 是一个迭代循环的过程：

- 拟定关于数据的问题；
- 通过对数据做可视化、变换、建模，得到问题答案；
- 利用得到的结果，重新改进问题，并（或）拟定新的问题。

[1] John Tukey: 近似地回答一个正确的问题（通常是模糊的问题），要比精确地回答一个错误问题（总是很清晰），要好的多.

EDA 的开始阶段，应该随意地研究你所能想到的各种想法（成功或失败均有可能），随着探索的推进，你将到达那些包含有效信息的位置，深入研究得到你想要的结果。

探索性数据分析通常包括：数据清洗、数据描述与汇总、数据变换、探索变量间的关系等。

希望读者能够通过探索性数据分析培养对数据的直觉。

数据清洗

数据模型结果的好坏很大程度上依赖于数据质量，很多数据集存在数据缺失、数据格式不统一、错误数据等情况，这就需要做**数据清洗。**

数据清洗通常包括：缺失值处理、数据去重、异常值处理、逻辑错误检测、数据均衡检测、处理不一致数据、相关性分析（剔除与问题不相关的冗余变量)、数据变换（标准/归一化、线性化、正态化等)。

数据清洗，常常占据了数据挖掘/机器学习的 70-80% 的工作量。

一. 缺失值

缺失值用 NA 表示，表明有值且占位的，只是该值是缺失值。区分：NULL 表示空值，不知道是否有值且不占位。

有的数据人为记录用特殊值或特殊符号代替缺失值，首先要替换成 NA:

```
replace_with_na(df, replace = list(x = 9999))  # naniar 包
```

按 R 的语法规则，NA 具有传染性，即有 NA 参与的计算，结果也是 NA. 所以很多 R 函数都带有参数 na.rm, 设置在计算时是否移除 NA.

```
mean(c(1,2,NA,4))
#> [1] NA
mean(c(1,2,NA,4), na.rm = TRUE)
#> [1] 2.33
```

1. 探索缺失值：naniar 包

(1) 缺失模式

缺失模式是描述缺失值与观测变量间可能的关系。从缺失的分布来讲，缺失值可以分为：

- **完全随机缺失** (MCAR)：某变量缺失值的出现完全是随机事件，与该变量自身无关，也与其他变量无关
- **随机缺失** (MAR)：某变量出现缺失值的可能性，与该变量自身无关，但与某些变量有关
- **非随机缺失** (MNAR)：某变量出现缺失值的可能性只与自身有关

若数据是 MCAR 或 MAR，则可以用相应的插补方法来处理缺失值；若数据是 MNAR，则问题比较严重，需要去检查数据的收集过程并试着理解数据为什么会丢失。

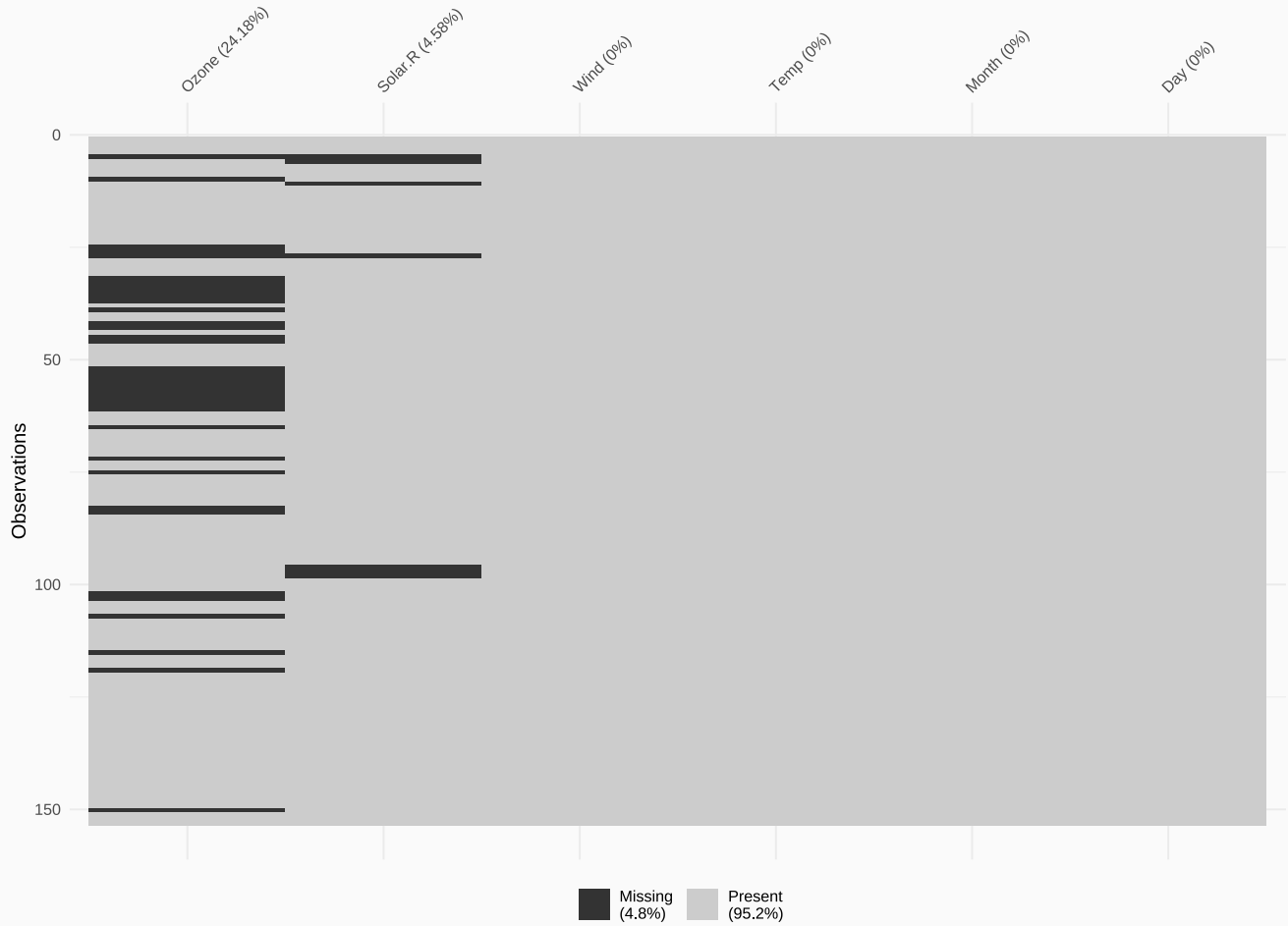
- `mcar_test()` 对数据进行 Little's MCAR 检验：

```
library(naniar)          # 探索与可视化缺失
mcar_test(airquality)    # 自带的空气质量数据集
#> # A tibble: 1 x 4
#>   statistic    df p.value missing.patterns
#>       <dbl> <dbl>   <dbl>            <int>
#> 1      35.1    14 0.00142                4
```

p 值 =0.00142<0.05, 拒绝原假设，故该数据不是 MCAR.

对于探索 MAR，函数 `vis_miss()` 可视化整个数据框，提供数据缺失的汇总信息：

```
vis_miss(airquality)
```



可见，变量 Ozone 和 Solar.R 有最多的缺失值，其他变量基本没有缺失。

(2) 缺失值统计

- 缺失数与缺失比

```
n_miss(airquality)                  # 缺失样本的个数
n_complete(airquality)              # 完整样本的个数
prop_miss_case(airquality)          # 缺失样本占比
prop_miss_var(airquality)           # 缺失变量占比
```

注：上述函数也接受向量，即判断数据框的某列。

- 样本（行）缺失汇总

```
miss_case_summary(airquality)   # 每行缺失情况排序
#> # A tibble: 153 x 3
#>    case n_miss pct_miss
#>   <int>  <int>    <dbl>
#> 1     5      2     33.3
#> 2    27      2     33.3
#> 3     6      1     16.7
#> # ... with 150 more rows
```

说明：第 5 行，缺失 2 个，缺失比例为 33.3%

```
miss_case_table(airquality)    # 行缺失汇总表
#> # A tibble: 3 x 3
#>   n_miss_in_case n_cases pct_cases
#>            <int>   <int>     <dbl>
#> 1              0     111     72.5
#> 2              1      40     26.1
#> 3              2       2      1.31
```

说明：缺失 0 个的行有 111 个，占比为 72.5%

- 变量（列）缺失汇总

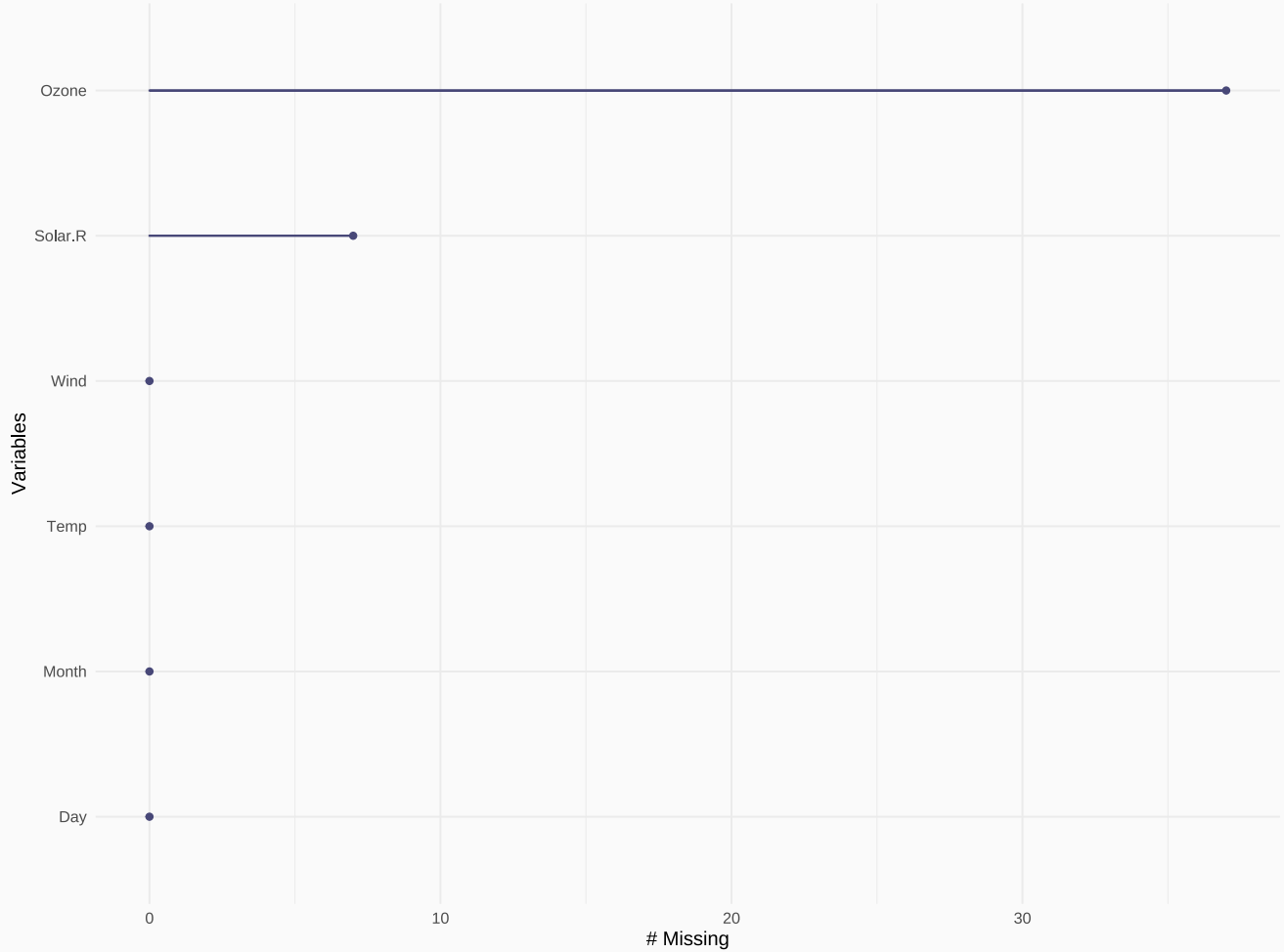
```
miss_var_summary(airquality)   # 每个变量缺失情况排序
#> # A tibble: 6 x 3
#>   variable n_miss pct_miss
#>   <chr>     <int>    <dbl>
#> 1 Ozone        37    24.2
#> 2 Solar.R       7     4.58
#> 3 Wind          0     0
#> # ... with 3 more rows
```

```
miss_var_table(airquality)     # 变量缺失汇总表
#> # A tibble: 3 x 3
#>   n_miss_in_var n_vars pct_vars
#>           <int>  <int>    <dbl>
#> 1             0      4     66.7
#> 2             7      1     16.7
#> 3            37      1     16.7
```

注：缺失汇总函数，还可以与 group_by() 连用，探索分组缺失情况。

- 缺失汇总函数，都有对应的可视化函数，比如

```
gg_miss_var(airquality)
```



(3) 对比缺失与非缺失

借助**影子矩阵**：与数据集同维数，标记各个数据是否缺失的矩阵，缺失表示为NA，而不缺失表示为!NA

bind_shadow()将影子矩阵按列合并到数据集，就可以分组汇总或绘图，以对比缺失与非缺失数据：

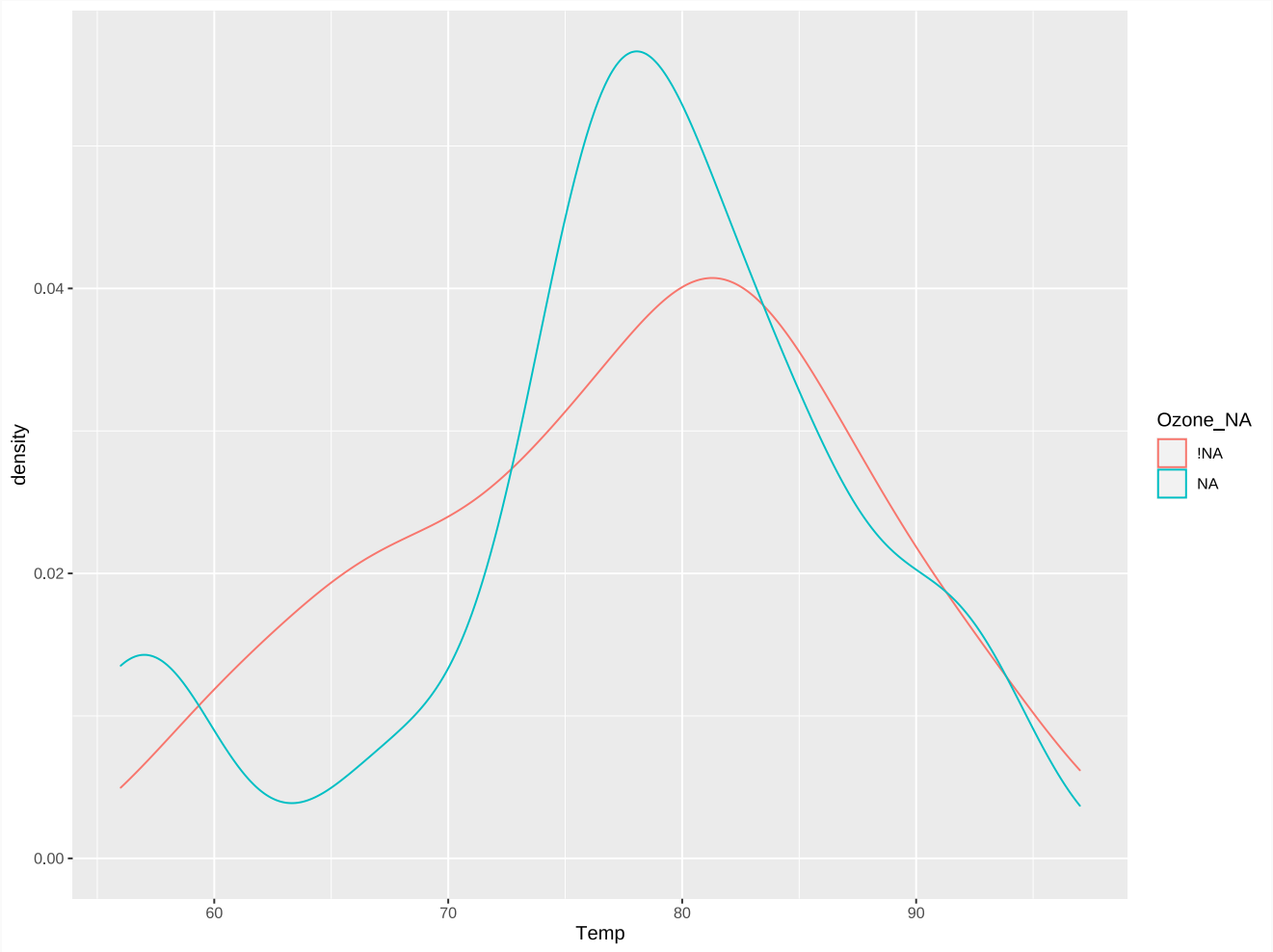
```
aq_shadow = bind_shadow(airquality)
aq_shadow
#> # A tibble: 153 x 12
#>    Ozone Solar.R  Wind  Temp Month   Day Ozone_NA Solar.R_N
#>    <int>   <int> <dbl> <int> <int> <int> <fct>    <fct>
#> 1     41     190   7.4    67     5     1 !NA      !NA
#> 2     36     118   8      72     5     2 !NA      !NA
#> 3     12     149  12.6    74     5     3 !NA      !NA
#> # ... with 150 more rows, and 2 more variables: Month_NA <
```

- 根据 Ozone 是否缺失，计算 Solar.R 的均值、标准差、方差、最小值和最大值：

```
aq_shadow %>%
  group_by(Ozone_NA) %>%
  summarise_at(.vars = "Solar.R",
               .funs = c("mean", "sd", "var", "min", "max"),
#> # A tibble: 2 x 6
#>   Ozone_NA  mean    sd   var   min   max
#>   <fct>    <dbl> <dbl> <dbl> <int> <int>
#> 1 !NA       185.  91.2 8309.     7   334
#> 2 NA        190.  87.7 7690.    31   332
```

- 根据 Ozone 是否缺失，绘制温度的分布图：

```
ggplot(aq_shadow, aes(Temp, color = Ozone_NA)) +
  geom_density()
```



2. 插补缺失值：simputation 包

- 若样本数据足够，缺失样本比例较小，可以直接剔除包含 NA 的样本：

```
na.omit(df)
```

- 若只想剔除某些列包含 NA 的行：

```
drop_na(df, <tidy-select>)
```

- 若想只剔除包含较多 NA 的行或列：

```
# 删除缺失超过 60% 的行
df %>%
  filter(pmap_lgl(., ~ mean(is.na(c(...))) < 0.6))
# 删除缺失超过 60% 的列
df %>%
  select(where(~ mean(is.na(.x)) < 0.6))
```

(1) 单重插补

`simputation` 包提供了许多常用的单重插补方法，每种方法都具有相似且简单的接口，支持：

- **基于模型 (可选增加随机误差)：**线性回归、稳健线性回归、岭/弹性网/Lasso 回归、CART 模型（决策树)、随机森林
- **多变量插补：**基于期望最大算法插补、缺失森林（迭代的随机森林插补）
- **其他方法：**（逐组）中位数插补（可选随机误差)、代理插补（复制另一个变量或使用简单变换来计算插补值)、应用为插补训练的模型

想要可视化查看插补效果，可以再结合 `naniar` 包。

通用插补语法：

```
impute_<模型>(data, formula, [模型设定选项])
```

返回结果类似 data 参数，除非空值被设定模型插补；formula 设定要插补的变量；模型设定是针对以及可能的数据集分组；公式的一般结构：

```
IMPUTED ~ MODEL_SPECIFICATION [| GROUPING]
```

其中，[ ] 为可选项。

- 用均值/中位数插补，适合连续变量，例如分组均值/中位数插补：

```
airquality %>%
  group_by(Month) %>%
  mutate(Ozone = naniar::impute_mean(Ozone))
impute_median(airquality, Ozone ~ Month)
```

- 用众数插补，适合分类变量。借用 get_mode() 计算每一列的众数替换该列的缺失值，若某列有多个众数，取第 1 个

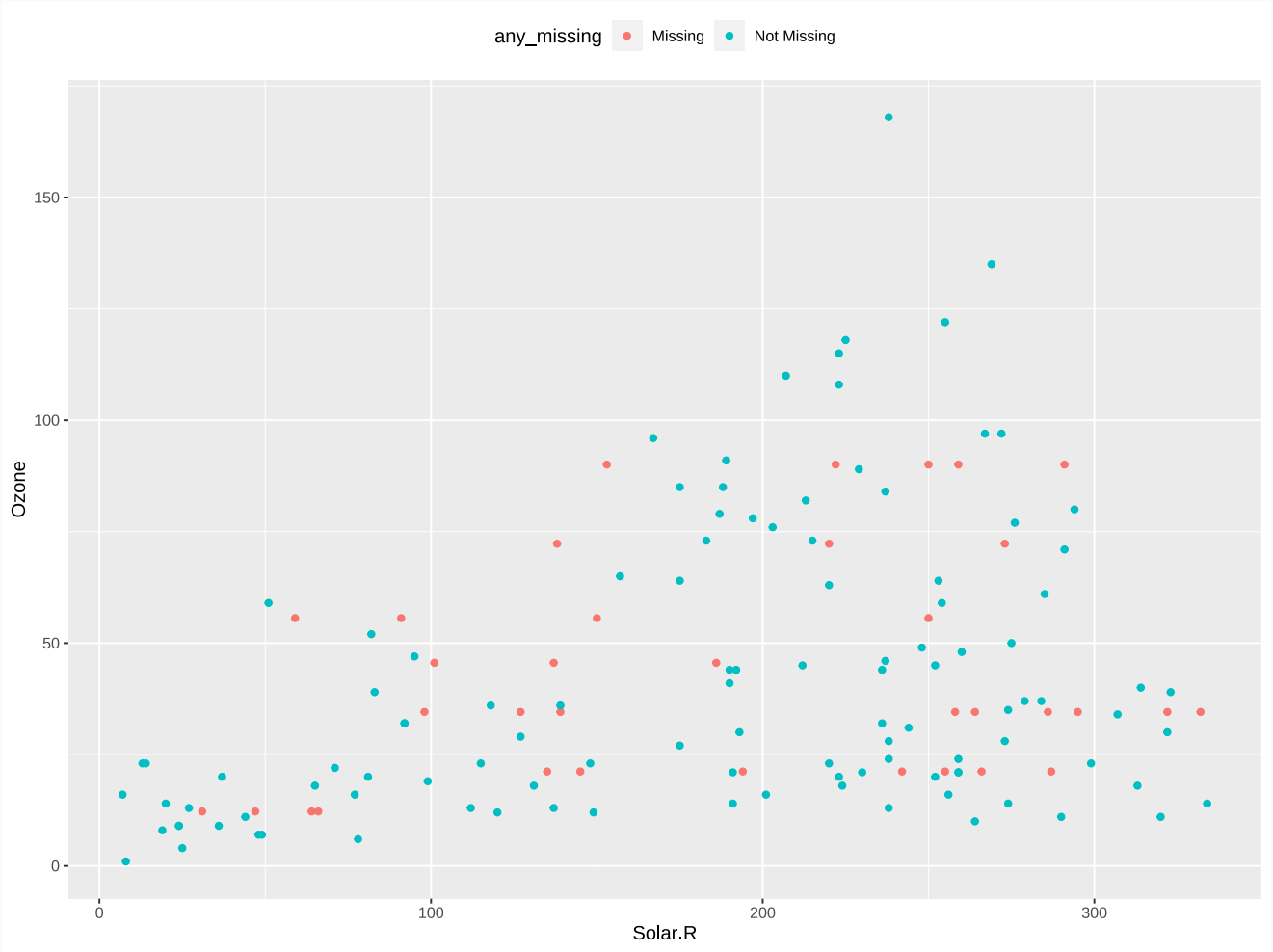
```
df %>%
   select(<tidy-select>) %>%    # 选择要插补的分类变量列
   map_dfc(~ replace_na(.x, rstatix::get_mode(.x)[1]))
```

- 用线性回归模型插补，即做插补变量关于其他变量的线性回归预测缺失值

```
impute_lm(airquality, Ozone ~ Solar.R + Wind + Temp,
          add_residual = "normal")  # 添加随机误差
```

- 用其他模型插补（需要相应的包），用法完全类似：
  - `impute_rlm()`: 用稳健线性回归模型插补
  - `impute_en()`: 用正则化线性回归模型插补
  - `impute_knn()`: 用 `k` 近邻模型插补，可设置邻居数参数 `k`
  - `impute_cart()`: 用决策树模型插补，可设置复杂度参数 `cp`
  - `impute_rf()`: 用随机森林模型插补，可设置复杂度参数 `cp`
  - `impute_mf()`: 用缺失森林模型插补
  - `impute_em()`: 用期望最大算法插补

```r
library(simputation)    # 单重插补
airquality %>%
  bind_shadow() %>% as.data.frame() %>%
  impute_cart(Ozone ~ Solar.R + Wind + Temp) %>%
  add_label_shadow() %>%
  ggplot(aes(Solar.R, Ozone, color = any_missing)) +
  geom_point() +
  theme(legend.position = "top")
```


any_missing
Missing
Not Missing
Ozone
150
100
50
0
0
100
200
300
Solar.R

(2) 多重插补

单重插补，就是只插补一次；而多重插补是插补多次：

- 将缺失数据集复制几个副本
- 对每个副本数据集进行缺失值插补
- 对这些插补数据集进行评估整合得到最终完整数据集

先用 mice 包的 mice() 函数实现多重插补：

```
library(mice)       # 多重插补
aq_imp = mice(airquality, m = 5, maxit = 10, method = "pmm",
              seed = 1, print = FALSE) # 设置种子, 不输出过程
```

- 参数 m 设置生成几个数据集副本；maxit 设置在每个插补数据集上的最大迭代次数；method 设置插补方法，针对连续、二分类、多分类变量的默认方法分别是 pmm，logreg，polyreg。

再用 complete() 函数获取经多重插补并整合的完整数据：

```
aq_dat = mice::complete(aq_imp)
```

另外，mice 包还提供了函数 with() 在每个插补数据集上建模分析，pool() 组合各个建模分析结果。

(3) 插值法插补

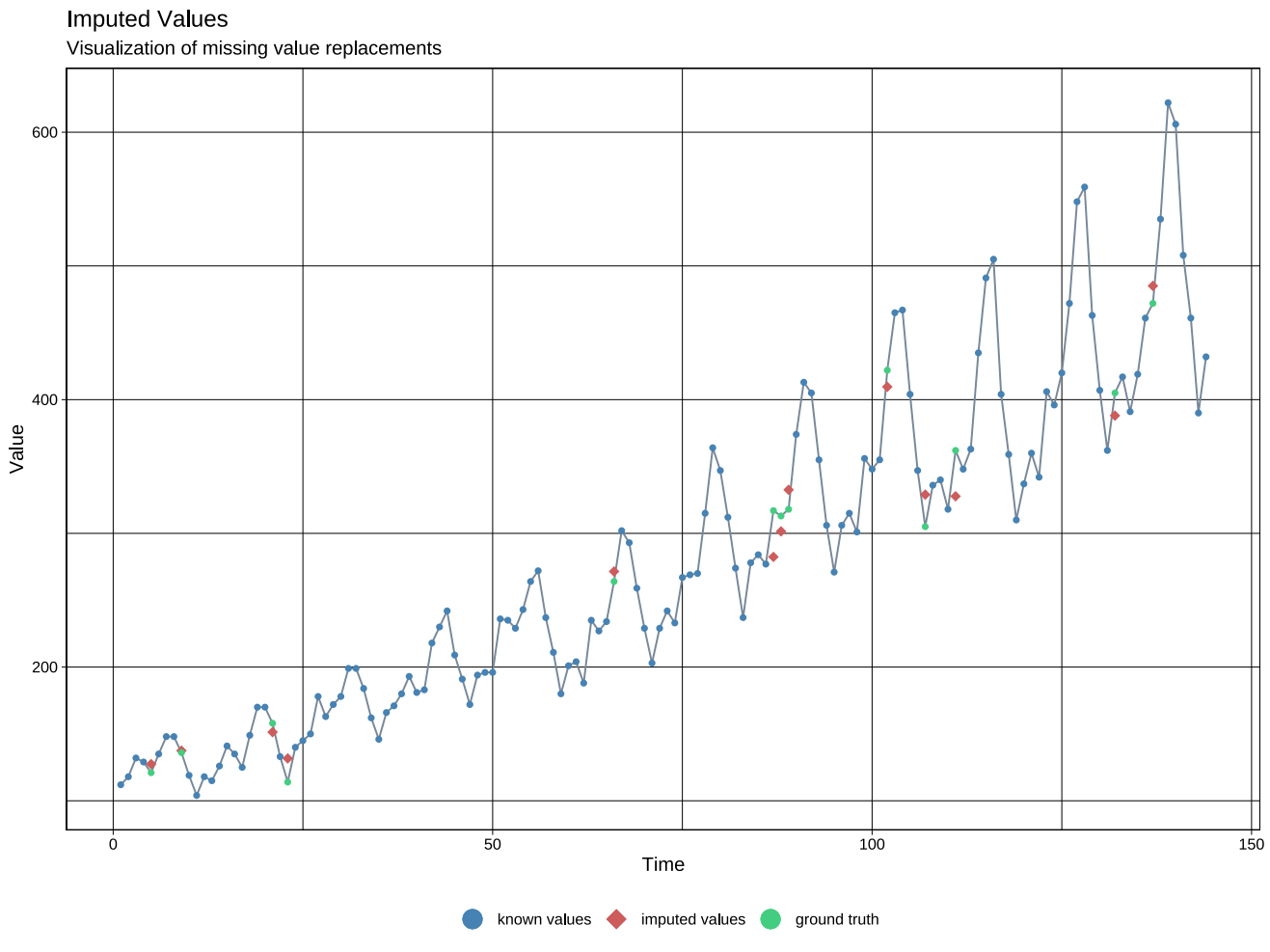
imputeTS 包实现了一系列插补和可视化时间序列数据的方法，包括插值法、时间序列分析算法等。

函数 na_interpolation() 可实现插值法插补，其参数 option 设置插值算法：linear (线性)，spline (样条)，stine(Stineman).

```
library(imputeTS)      # 插补时间序列
imp = na_interpolation(tsAirgap, option = "spline")
```

其他插补函数还有 na_kalman()(Kalman 光滑), na_ma() (指数移动平均), na_seadec() (季节分解) 等。

```
ggplot_na_imputations(tsAirgap, imp, tsAirgapComplete)
```



二. 异常值

异常值，是指与其他值或其他观测相距较远的值或观测，即与其他数据点有显著差异的数据点。异常值会极大地影响模型的效果。

数据预处理，包括异常值的检测与处理（直接剔除或替换为 NA 再插补）。另外，异常值检测也可能是研究目的，例如识别数据造假、交易异常等。

1. 单变量的异常值

(1) 标准差法

若数据近似正态分布，则大约 68%的数据落在均值的 1 个标准差之内，大约 95%落在 2 个标准差之内，而大约 99.7%落在 3 个标准差之内。如果数据点落在 3 倍标准差之外，则认为是异常值。

(2) 百分位数法

基于百分位数，所有落在 2.5 和 97.5 百分位数 (也可以是其他百分位数) 之外的数据都认为是异常值。

(3) 箱线图法

箱线图的主要应用之一就是识别异常值。以数据的上下四分位数 (Q1-Q3) 为界画一个矩形盒子（中间 50% 的数据落在盒内)，盒长为 IQR = Q3-Q1, 默认盒须不超过盒长的 1.5 倍，之外的点认为是异常值。

```r
univ_outliers = function(x, method = "boxplot", k = NULL,
                         coef = NULL, lp = NULL, up = NULL){
  switch(method,
    "sd" = {
       if(is.null(k)) k = 3
       mu = mean(x, na.rm = TRUE)
       sd = sd(x, na.rm = TRUE)
       LL = mu - k * sd
       UL = mu + k * sd},
    "boxplot" = {
       if(is.null(coef)) coef = 1.5
       Q1 = quantile(x, 0.25, na.rm = TRUE)
       Q3 = quantile(x, 0.75, na.rm = TRUE)
```

```
        iqr = Q3 - Q1
        LL = Q1 - coef * iqr
        UL = Q3 + coef * iqr},
    "percentiles" = {
        if(is.null(lp)) lp = 0.025
        if(is.null(up)) up = 0.975
        LL = quantile(x, lp)
        UL = quantile(x, up)
  })
  idx = which(x < LL | x > UL)
  n = length(idx)
  list(outliers = x[idx], outlier_idx = idx, outlier_num = n)
}
```

参数说明：

- `x` 为数据向量
- `method` 选择识别异常值的方法: “boxplot”（默认）, “sd”, “percentiles”
- `k` 配合”sd” 法, 设置均值加减标准差的倍数, 默认为 `3`
- `coef` 配合”boxplot” 法, 设置盒须长度关于 `IQR` 的倍数, 默认为 `1.5`
- `lp` 和 `up` 配合”percentiles” 法, 设置百分位数下限和上限, 默认为 `0.025` 和 `0.975`

```r
x = mpg$hwy
univ_outliers(x)                    # 箱线图法
#> $outliers
#> [1] 44 44 41
#>
#> $outlier_idx
#> [1] 213 222 223
#>
#> $outlier_num
#> [1] 3
univ_outliers(x, method = "sd")  # 标准差法
#> $outliers
#> [1] 44 44
#>
#> $outlier_idx
#> [1] 213 222
#>
```

(1) 局部异常因子法 (LOF) 法

LOF 法是基于概率密度函数识别异常值的算法，其原理是：将一个点的局部密度与其周围点的密度相比较，若前者明显的比后者小 (LOF 值大于 1)，则该点相对于周围的点来说就处于相对比较稀疏的区域，这就表明该点是异常值。

LOF 法可以用 DMwR2 包中的 lofactor() 函数实现，Rlof 包中函数 lof() 可实现相同的功能，并且支持并行计算和选择不同距离。

```
library(DMwR2)
lofs = lofactor(iris[,1:4], k = 10)  # k 为邻居数
# 选择 LOF 值最大的 5 个索引，认为是异常样本
order(lofs, decreasing = TRUE)[1:5]
#> [1]  42 107  23  16  99
```

(2) 基于聚类算法

通过把数据聚成类，将那些不属于任何一类的数据作为异常值。

DMwR2 包提供了 outliers.ranking() 函数，基于层次聚类来计算的异常值值的概率及排名，具体是根据聚合层次聚类过程的各个样本的合并路径来获得排名。

```
rlt = outliers.ranking(iris[,1:4])
# rlt$rank.outliers[1:5]      # 异常值排名前五的样本
sort(rlt$prob.outliers, decreasing = TRUE)[1:5]
#>    36    42   107    58    61
#> 0.818 0.800 0.800 0.692 0.692
```

也可以借助其他聚类算法包（dbscan，stats）做聚类分析，再进一步筛选出异常值：

- 基于密度的聚类 DBSCAN，如果对象在稠密区域紧密相连，则被分组到一类；那些不会被分到任何一类的对象就是异常值；
- 基于 k-means 聚类，将数据分成 k 组，通过把它们分配到最近的聚类中心；再计算每个样本到聚类中心的距离（或相似性），并选择距离最大的若干样本作为异常值。

(3) 基于模型的异常值

在对回归模型做模型诊断时会做强影响分析：通常回归模型得具有一定的稳定性，若加入和移出某个样本进入对模型有着巨大影响，则该样本是应该剔除的异常值。

度量这种强影响的指标有：

- Cook’s 距离：`cooks.distance(model)`
- Leverage 值：`hatvalues(model)`

或者用 `influence.measures(model)` 直接计算包括二者在内的 4 个强影响度量值。

另外，car 包中提供了 Bonferroni 异常值检验函数 `outlierTest(model)`，支持线性回归、广义线性回归、线性混合模型。

```r
mod = lm(mpg ~ wt, mtcars)
car::outlierTest(mod)
#> No Studentized residuals with Bonferroni p < 0.05
#> Largest |rstudent|:
#>          rstudent unadjusted p-value Bonferroni p
#> Fiat 128     2.54              0.0168         0.537
```

结果表明，行名为 Fiat 128 的样本是异常值。

(4) 随机森林法异常值检测

相当于是单变量标准差法异常值检测的多变量版本，但看的不是偏离全局均值，而是偏离条件均值多远，而条件均值是基于随机森林计算的。

每个数值变量，都基于其他变量做随机森林回归，若观测值与”袋外”预测值的标准化绝对偏差大于 3 倍的”袋外”预测值的 RMSE，则认为是异常值。这样识别出来的异常值，可以用非异常值预测的均值替换。

outForest 包提供了 outForest(data, formula, replace, ...) 函数：

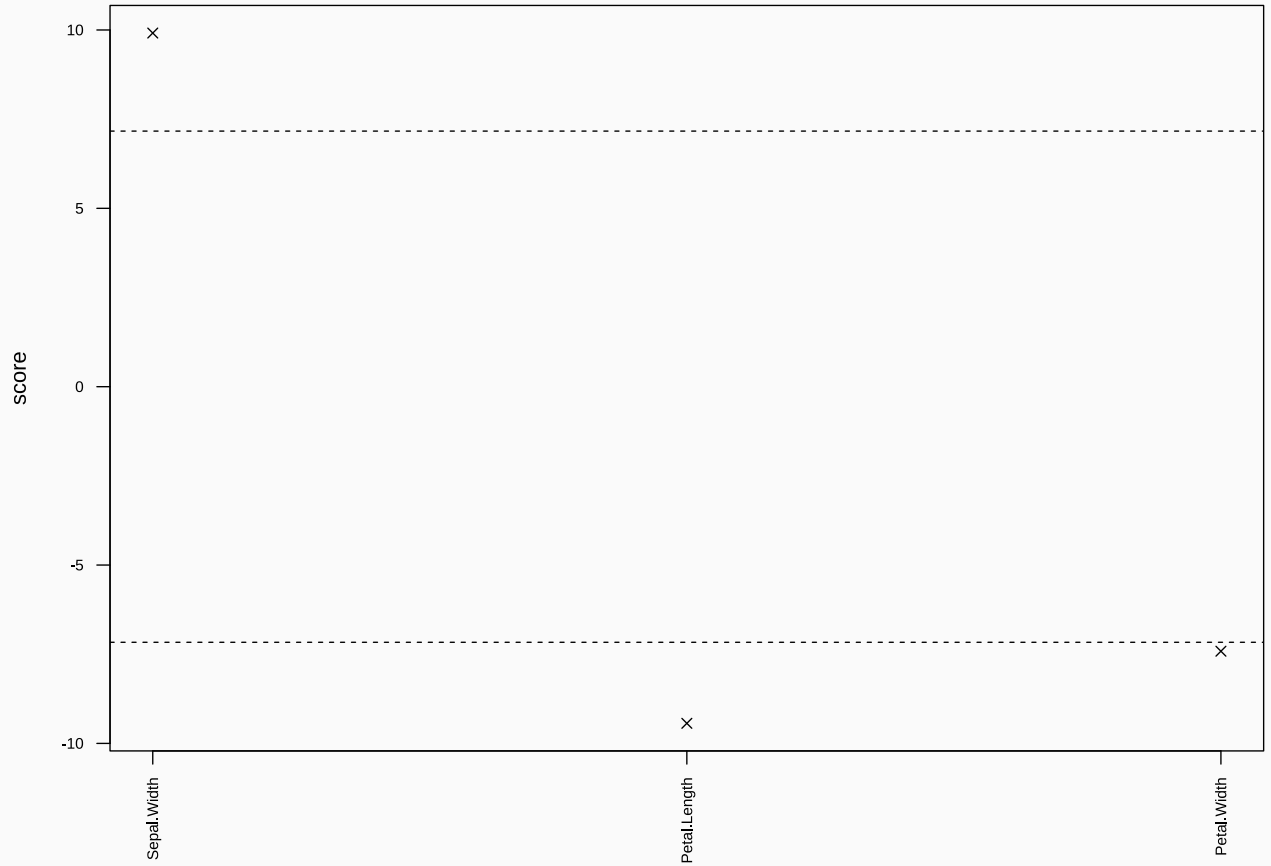
- data 为数据框
- formula = .~. 设置公式，默认是用右侧所有变量检测左侧所有数值变量；
- replace 设置如何替换异常值，可选"pmm", "predictions", "NA", "no", 插补值是基于 missRanger::missRaner() 生成的预测值；

其他参数，可设置保留多少异常值，控制随机森林复杂度[2]等。

[2]可用来提速的随机森林复杂度参数: num.trees, mtry, sample.fraction, max.depth, min.node.size.

```r
library(outForest)
# 用 iris 数据随机生成若干异常值
irisWithOut = generateOutliers(iris, p = 0.02, seed = 123)
# 检测除 Sepal.Length 外数值变量异常值, 异常值数设为 3
out = outForest(irisWithOut, . - Sepal.Length ~ .,
                max_n_outliers = 3, verbose = 0)
outliers(out)                    # 查看异常值及相关信息
#>   row           col observed predicted  rmse score threshol
#> 1  72  Sepal.Width    11.50      2.68 0.890  9.91       7.1
#> 2 103 Petal.Length   -14.00      6.05 2.125 -9.44       7.1
#> 3  53  Petal.Width    -1.94      1.60 0.477 -7.42       7.1
```

```
plot(out, what = "scores")  # 绘制各变量异常值得分图
```



要取出替换异常值之后的数据，用 `Data(out)`.

最后注：还有 `rstatix` 包提供了 `mahalanobis_distance()` 函数计算多变量的马氏距离，进而标记基于马氏距离的异常值；`anomalize` 包检验时间序列的异常值；`outliers` 包提供了一系列专用的检验异常值的函数。

本篇主要参阅 (张敬信, 2022) 以及包文档，模板感谢 (黄湘云, 2021), (谢益辉, 2021).

参考文献

张敬信 (2022). *R* 语言编程：基于 *tidyverse*. 人民邮电出版社, 北京.

谢益辉 (2021). *rmarkdown: Dynamic Documents for R*.

黄湘云 (2021). *Github: R-Markdown-Template*.