УДК 004.054

Руденко І.І.,
Моденов Ю.Б., к.т.н.

DATA TIDYING: ПІДГОТОВКА СТАТИСТИЧНИХ ДАНИХ НА МОВІ R

Національний авіаційний університет
modenov1951@gmail.com

Є поширена думка, що до 80% процесу аналізу даних – це час, витрачений на їх нормалізацію. У цій статті ми зупинимося лише на одному аспекті підготовки даних до аналізу – структурування та впорядкування наборів даних – data tidying

Ключові слова: набори даних, нормалізація, лістинг, функція Gather.

Вступ

Процес "*data tidying*" передує безпосередній обробці даних і є особливо корисним інструментом для тих, хто хоче проаналізувати поведінку користувачів не застосовуючи дорогі засоби аналізу такі як *Intercom*. Таким чином, ми можемо деінкрементувати два важливі економічні показники як вартість кінцевого продукту та час, що затрачений на розробку бази даних. Перекласти термін «*tidy data*» на українську мову можна як «впорядковані дані». Варто розпочати із питання, що таке упорядкований набір даних.

Набори даних

Велика частина наборів даних - це таблиці, що містять рядки і стовпці. Набори даних містять значення. Зазвичай це або цифри (кількісні дані) або рядки (якісні дані). Кожне значення ставиться з одного боку до змінної, з іншого боку до відповідного спостереження. При цьому спостереження можуть групуватися в типи одиниць спостереження (*observation units*), щоб забезпечити їх роздільне зберігання і не допустити можливих невідповідностей.

Набір даних може бути впорядкованим або хаотичним в залежності від того, як рядки, стовпці і таблиці відповідають спостереженням, змінним і типам одиниць спостереження.

Є три ознаки упорядкованого набору даних:

– кожна змінна формує стовпець;

– кожне спостереження формує рядок;

– кожен тип одиниці спостереження формує таблицю.

Порушення кожного з перерахованих ознак означає що набір даних є хаотичним або невпорядкованим.

Правила впорядкування набору статистичних даних

У реальному світі отримати відразу упорядкований набір даних ви можете, хіба що, випадково. Розглянемо п'ять основних проблем в наборах даних і шляхи їх вирішення за допомогою пакетів мови *R tidyr і dplyr*.

1. У заголовках стовпців знаходяться значення, а не імена змінних. Розглянемо масив даних *students*

```
> students
  grade male female
1     A    1      5
2     B    5      0
3     C    5      2
4     D    5      5
5     E    7      4
```

Лістинг 1. Масив даних students в середовищі R

У першій колонці *Grade* вказані оцінки, які отримали студенти, а в другій і третій – яка кількість хлопців і дівчат відповідно серед них було. Насправді в цьому наборі даних три змінних - оцінка, стать і кількість. Значення змінної «стать» міститься в заголовках другого і третього стовпця. Змінна кількість описує, скільки

є студентів для кожної комбінації оцінки і статі.

Щоб упорядкувати цей набір даних, нам потрібно досягти такого результату, щоб кожен стовпець описував окрему змінну. Це можна легко зробити за допомогою функції *Gather*:

```
> gather(students, sex, count, -grade)
  grade   sex count
1     A  male     1
2     B  male     5
3     C  male     5
4     D  male     5
5     E  male     7
```

Лістинг 2. Впорядкований масив даних students за допомоги функції Gather

За допомогою функції *Gather* ми збираємо декілька стовпців в пари key-*value*. В даному випадку *sex* – це *key*, а *count - value*. Параметр «*-grade*» означає, що ця змінна в процесі не бере участь і залишається без змін.

2. Кілька змінних зберігаються в одному стовпці

Розглянемо набір даних Students2:

```
> students2
  grade male_1 female_1 male_2 female_2
1     A      3        4      3        4
2     B      6        4      3        5
3     C      7        4      3        8
4     D      4        0      8        1
5     E      1        1      2        7
```

Лістинг 3. Масив даних students2

Він схожий на попередній набір даних. Відмінність полягає в тому, що тут є поділ за двома класами і кількість студентів вказано з розподілом за статтю та класом. До того ж тут додаються нові змінні «стать» і «клас», що зберігаються в одному стовпці. В даному випадку проблема впорядкування набору даних вирішується в два кроки. Спочатку виводимо змінну *count*, зберігаючи об'єднання змінних статі і класу. Далі розділяємо змінні «стать» і «клас» по різних стовпцях. Для зручності в міні-скрипті з'єднуємо всі дії оператором *chain*:

```
students2 %>%
  gather(sex_class, count, -grade) %>%
  separate(sex_class, into = c("sex", "class")) %>%
  print
```

Лістинг 4. Скрипт на мові R, що об'єднує змінні статі та класу оператором chain

3. Змінні зберігаються як в стовпцях, так і в рядках

Розглянемо масив даних students3:

```
> students3
   name    test class1 class2 class3 class4 class5
1 Sally midterm      A   <NA>      B   <NA>   <NA>
2 Sally   final      C   <NA>      C   <NA>   <NA>
3  Jeff midterm   <NA>      D   <NA>      A   <NA>
4  Jeff   final   <NA>      E   <NA>      C   <NA>
5 Roger midterm   <NA>      C   <NA>   <NA>      B
```

Лістинг 5. Масив даних students3

Для кожного з п'яти студентів у нас є проміжна і фінальна оцінка. При цьому кожен навчався у двох класах з п'яти можливих. Проблеми цього набору даних починаються з того, що імена стовпців *class1: class5* містять значення однієї змінної *class*. Значення стовпця *test (midterm, final)* повинні бути змінними і містити значення *grade* для кожного студента.

Для вирішення цього завдання організуємо змінну *class* і помістимо імена стовпців *class1: class5* в її значення. Далі розкриємо значення стовпця *test* в змінні *final і midterm*. І, нарешті, заберемо ентропію значень змінної клас, залишивши там тільки цифри. Нижче наведено міні-скрипт:

```
students3 %>%
  gather(class, grade, class1:class5, na.rm = TRUE) %>%
  spread(test, grade) %>%
  mutate(class, class = extract_numeric(class)) %>%
  print
```

Лістинг 6. Скрипт на мові R, що організовує змінну class з оператором mutate

Результат його роботи - впорядкований масив даних:

```
  name      class  final midterm
1  Brian      1     B       B
2  Brian      5     C       A
3   Jeff      2     E       D
4   Jeff      4     C       A
5  Karen      3     C       C
```

Лістинг 6. Скрипт на мові R, що організовує змінну class з оператором mutate

4. Кілька типів одиниць спостереження зберігаються в одній таблиці

Наступний набір даних *students4* виглядає практично так само, як упорядкований *students3* з попереднього прикладу. Головна відмінність - додалися стовпці *id* і стать.

```
> students4
   id    name sex class midterm  final
1 168   Brian   F     1       B     B
2 168   Brian   F     5       A     C
3 588   Sally   M     1       A     C
4 588   Sally   M     3       B     C
5 710    Jeff   M     2       D     E
```

Лістинг 7. Масив даних students4

Проблема набору полягає в надмірності даних - поєднання (*id, name, sex*) зустрічаються по два рази. Рішення завдання полягає в поділі набору даних за двома таблицями:

– у першій буде зберігатися інформація по студентам (*id, name, sex*);

– у другій зберігатиметься інформація за оцінками (*id, class, midterm, final*)

Збираємо таблицю інформації по студентам (вибираючи потрібні стовпці і видаляючи дублікати). Далі робимо вибірку необхідних нам полів. В результаті отримаємо наступний впорядкований масив даних:

```
gradebook <- students4 %>%
  select(id, class, midterm, final) %>%
  print
  id class midterm final
1 168     1       B     B
2 168     5       A     C
3 588     1       A     C
4 588     3       B     C
5 710     2       D     E
```

Лістинг 8. Видаляємо дублікати і отримуємо новий впорядкований масив

5. Одна одиниця спостереження зберігається в кількох таблицях

Наведемо приклад зворотний попередньому - одиниця спостереження зберігається в різних таблицях (*passed* - ті, хто іспит здав і *failed* - ті, хто не здав):

```
> passed
  name    class final
1 Brian       1     B
```

```
2 Roger     2   A
3 Roger     5   A
4 Karen     4   A
> failed
   name  class final
1 Brian     5   C
2 Sally     1   C
3 Sally     3   C
4  Jeff     2   E
5  Jeff     4   C
6 Karen     3   C
```

Лістинг 10. Впорядкований масив за ознакою статусу проходження іспиту

При цьому інформація про те, чи здав студент іспит чи ні, міститься в самій оцінці (*A, B* - здав, в інших випадках - не здав).

Об'єднаймо інформацію в одну таблицю, попередньо додавши стовпець *status* з ознакою, чи здав студент іспит чи ні:

```
passed <- mutate(passed, status = "passed")
failed <- mutate(failed, status = "failed")
> rbind_list(passed, failed)
Source: local data frame [10 x 4]

   name  class final status
1 Brian     1     B passed
2 Roger     2     A passed
3 Roger     5     A passed
4 Karen     4     A passed
5 Brian     5     C failed
6 Sally     1     C failed
7 Sally     3     C failed
8  Jeff     2     E failed
9  Jeff     4     C failed
10 Karen    3     C failed
```

Лістинг 11. Кінцевий результат - впорядкований масив students

Висновок

В результаті отримали упорядкований набір даних з ознакою *passed / failed*, використавши при цьому пакети мови R за допомоги методології *data tidying*. Наукове значення даної методології обробки великих масивів статистичних даних має великий влив на роботу аналітиків даних. Використовуючи алгоритми впорядкування інформації ми маємо змогу зекономити кошти та час на нормалізації розробленої бази даних.

Список літератури

1. Winston Chang. R Graphics Cookbook/Winston Chang. – Sebastopol: O'Reilly Media, 2013. – 253 ст. ISBN: 978-1-4493-1695-2 | ISBN 10:1-4493-1695-6.

2. Paul Teetor. R Cookbook/Paul Teetor. - Sebastopol: O'Reilly Media, 2011. – 438 ст. 978-0-596-80915-7 | ISBN 10:0-596-80915-8.

3. Hadley Wickham. Tidy Data [Електронний ресурс] – 2012. – Режим доступу: http://vita.had.co.nz/papers/tidy-data.pdf.

4. Habrahabr. Data Tidying: на конкретных примерах [Електронний ресурс] – 2013. – Режим доступу: http://habrahabr.ru/post/248741/.

Статтю подано до редакції 23.03.2015