homework v

Vivek Panchal,Rohit Kunjilikattil

2019-10-8

Introduction

In this assignment, we clean the NYC311 data to make sure it conforms to the tidy data standards. For this we use various tidyr functions like **separate** and **filter**. We transform the dataset into tidy data. Further we select another dataset, the nyc_payroll_information dataset, and clean that dataset so that it too follows the standards of tidy data. We have merged the two datasets using **Borough**.

Tidyverse is a r package that helps in making data tidy. Tidy data is defined as data which follows three basic rules : each variable must have its own column, each observation must have its own row and finally each value must have its own cell.One of the advantages of tidy data is it's easier to work with data that always has a consistent structure. Further, R's vectorised nature makes it particularly easy to transform tidy data.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
install.packages('tidyverse', repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/vivek 14/Documents/R/win-library/3.6'
## (as 'lib' is unspecified)

## Warning: package 'tidyverse' is in use and will not be installed
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(dplyr)
```

```
nyc311_a = fread("C:/Users/vivek 14/Downloads/311_Service_Requests_from_2010_to_Present.csv",nrows = 25
```

Cleaning Data

Here, we first try to clean data as much as possible so that we do not have to waste our processing power on data that we don't want or on bad data. As the complaint type is one of the most important columns, we try to make it as uniform as possible. We go about doing this by removing any unwanted spaces, replacing hyphens with slashes, etc. We then filter the zipcodes to only have values of length 5.

```
nyc311_a$Complaint.Type = tolower(nyc311_a$Complaint.Type)
nyc311_a$Complaint.Type = gsub('s$', '', nyc311_a$Complaint.Type)
nyc311_a$Incident.Zip = gsub('-[[:digit:]]{4}$', '',nyc311_a$Incident.Zip)
nyc311_a$Complaint.Type = gsub('paint - plaster', 'paint/plaster', nyc311_a$Complaint.Type)
nyc311_a$Complaint.Type = gsub('general construction', 'construction', nyc311_a$Complaint.Type)
nyc311_a$Complaint.Type = gsub('nonconst', 'construction', nyc311_a$Complaint.Type)
nyc311_a$Complaint.Type = gsub('street sign - [[:alpha:]]+', 'street sign', nyc311_a$Complaint.Type)
nyc311_a$Complaint.Type = gsub('fire alarm - .+','fire alarm', nyc311_a$Complaint.Type)
nyc311_clean2 = nyc311_a
```

Separating Columns

Here, we see multiple columns that are of class character that have date and time values together in one column. So we use the **separate** function from tidyr library to separate such columns into separate columns with date and time stored separately.

```
nyc311_clean2 <- nyc311_clean2 %>%
  separate(`Created Date`, into = c("Created Date", "Created Time"), sep = 10)

nyc311_clean2 <- nyc311_clean2 %>%
  separate(`Closed Date`, into = c("Closed Date", "Closed Time"), sep = 10)

nyc311_clean2 <- nyc311_clean2 %>%
  separate(`Due Date`, into = c("Due Date", "Due Time"), sep = 10)

nyc311_clean2 <- nyc311_clean2 %>%
  separate(`Resolution Action Updated Date`, into = c("Resolution Action Updated Date", "Resolution Acti
```

Removing unwanted columns

Further we remove rows which have 'Unspecified' listed for Boroughs. Finally as all of this data is from New York City, we remove the city column as it is redundant.

```
library(data.table)
nyc311_clean2 <- data.table(nyc311_clean2)
nyc311_clean2 <- nyc311_clean2[,c("City"):=NULL]
```

```
nyc311_clean2 <- nyc311_clean2 %>% filter(!str_detect(Borough, "Unspecified"))
nyc311nodups<-distinct(nyc311_clean2)
isTRUE(all.equal(nyc311nodups,nyc311_clean2))
```

```
## [1] FALSE
```

Reading a dataset from a url

Here we use r to read and download the csv file directly from the url. The saves us the trouble of sharing the csv file along with the rmd file. This way is much more efficient. We are using NYC payroll data. This dataset contians the salary, pay rate and total compensation of every New York City employees from year 2014 till 2017.This dataset provides columns for fiscal year, agency they work for, borough they are working in.This will provide us with an insight into who gets paid how much and for what. The source dataset that we are extracting from url already follows the rules of tidydata. Hence, we do not use tidyr functions to further clean the data.

```
nyc_payroll <- fread("https://data.cityofnewyork.us/api/views/k397-673e/rows.csv?accessType=DOWNLOAD",n
nycpayroll_nodups<-distinct(nyc_payroll)
isTRUE(all.equal(nycpayroll_nodups,nyc_payroll))
```

```
## [1] TRUE
```

There are 3 types of pay basis i.e Per hour, Per day and Per annum. We are trying to explore this by finding the total count of such employees working in New York city.

```
nyc_payroll %>% dplyr::group_by(`Pay Basis`) %>% dplyr::summarise(n = n())
```

```
## # A tibble: 3 x 2
##   `Pay Basis`     n
##   <chr>       <int>
## 1 per Annum   24700
## 2 per Day        58
## 3 per Hour      242
```

Here we have found out the total count of records by year.

```
nyc_payroll %>% dplyr::group_by(`Fiscal Year`) %>% dplyr::summarise(n = n())
```

```
## # A tibble: 4 x 2
##   `Fiscal Year`     n
##           <int> <int>
## 1          2015  2109
## 2          2016  6833
## 3          2018  7561
## 4          2019  8497
```

Showing first five records of the nyc_payroll dataset.

```
library(knitr)
nyc_payroll %>%
  head(5)%>%
  kable()
```

Fiscal Year	Payroll Number	Agency Name	Last Name	First Name	Mid Init	Agency S
2019	67	ADMIN FOR CHILDREN'S SVCS	SIMMONS	DONALD		07/04/20
2019	67	ADMIN FOR CHILDREN'S SVCS	MOHAMMED	KATHIE	S	10/24/20
2019	67	ADMIN FOR CHILDREN'S SVCS	MCRAE	TANESIA	M	09/11/20
2019	67	ADMIN FOR CHILDREN'S SVCS	ROZON	GINNETTE		08/14/20
2019	67	ADMIN FOR CHILDREN'S SVCS	LOPEZ	RAFAEL		01/17/20

Exploration of NYCPayroll dataset

In this part of analysis we have split our dataset into two parts, agencies that have income less than 10k and over 10k.

```
library(knitr)
library(dplyr)
byagencysub10<- nyc_payroll %>% count(nyc_payroll$`Agency Name`, nyc_payroll$`Fiscal Year`) %>% ungroup
kable(head(byagencysub10,10))
```

nyc_payroll$Agency Name	nyc_payroll$Fiscal Year	n
ADMIN FOR CHILDREN'S SVCS	2019	8497
ADMIN FOR CHILDREN'S SVCS	2018	7561
ADMIN FOR CHILDREN'S SVCS	2016	6833
ADMIN FOR CHILDREN'S SVCS	2015	2109

Here we have represented agencies that have **Base Salary** more than 10K. We have made use of kable() of knitr library to display our data in a concise manner

```
library(knitr)
library(dplyr)
byunion10plus<- nyc_payroll %>% count(nyc_payroll$`Agency Name`,nyc_payroll$`Title Description`,nyc_payr
kable(head(byunion10plus,10))
```

nyc_payroll$Agency Name	nyc_payroll$Title Description	nyc_payroll$Base Salary
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	53126
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	54720
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	57070
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	53519
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	51315
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	51830
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	47250
ADMIN FOR CHILDREN'S SVCS	YOUTH DEVELOPMENT SPECIALIST	44426
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST	49279
ADMIN FOR CHILDREN'S SVCS	CHILD PROTECTIVE SPECIALIST SUPERVISOR	80146

We calculated the average salary of each agency in terms of `Regular Gross Paid`. We made use of dplyr functions for this analysis. We have grouped by the `Fiscal year` column of nycpayroll dataset.Mutate() helped us in creating a new variable from the data. The summarise_at() affects variables selected with character vector or vars().vars() is nothing but a character vector of column names, a numeric vector of column positions.

```
library(knitr)
library(dplyr)

salbyagency <- nyc_payroll %>%
  group_by(`Fiscal Year`) %>%
  mutate(Count = n()) %>%
  group_by(`Agency Name`, `Count`) %>%
  summarise_at(vars(`Regular Gross Paid`), funs(mean(., na.rm = TRUE))) %>%
  arrange(desc(`Regular Gross Paid`))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```
kable(head(salbyagency, 10))
```

Agency Name	Count	Regular Gross Paid
ADMIN FOR CHILDREN'S SVCS	7561	55308.35
ADMIN FOR CHILDREN'S SVCS	8497	54712.02
ADMIN FOR CHILDREN'S SVCS	6833	53589.70
ADMIN FOR CHILDREN'S SVCS	2109	47749.09

In our dataset we have a column `Agency Start Date` so what we did is we created a new column `yrs` from it which has the year of that particular person when he started working. We made use of POSIXct for this purpose. Further to calculate experience we took out present date and from that we subtracted agency start date.We performed group by on Boroughs so that we get average years of experience of each Borough.

```
library(knitr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
```

```
##     hour, isoweek, mday, minute, month, quarter, second, wday,
##     week, yday, year

## The following object is masked from 'package:base':
##
##     date
```

```
library(dplyr)
nyc_payroll$`yrs` <- year(as.POSIXct(nyc_payroll$`Agency Start Date`, format="%d/%m/%Y"))
today <- format(Sys.Date(), "%Y")
nyc_payroll$exp <- as.numeric(today)-as.numeric(nyc_payroll$yrs)
salbyexp<- nyc_payroll%>%
  group_by(`Work Location Borough`) %>%
  select(`Work Location Borough`,exp) %>%
  dplyr::summarise_at(vars(`exp`), funs(mean(., na.rm=TRUE)))
kable(head(salbyexp,10))
```

Work Location Borough	exp
BRONX	7.390135
BROOKLYN	6.825243
MANHATTAN	9.773065
QUEENS	8.139726
RICHMOND	6.200000

As we are going to join our nyc payroll dataset with nyc311 open data set we have common columns such as Boroughs and Agency for that purpose we renamed our column name of `Work Location Borough` of nyc_payroll dataset to `Borough` so that we can perform merge operation.

```
library("magrittr")
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library("data.table")
nyc_pr_br<-as.data.table(nyc_payroll) %>% {setnames(., old = "Work Location Borough", new = "Borough")[]
```

We decided to join the databases upon the **Borough** column because it was the most obvious choice. The aim of this join was to add the average experience by borough calculated from the nyc_payroll database and use it to get the average experience for the agencies in the nyc311. The joined table contains the Agency name from the nyc311 database, the exp column from the nyc_payroll dataset and Borough which a common column, on which the join was performed.

```
nyc_pr_exp<-as.data.table(salbyexp) %>% {setnames(., old = "Work Location Borough", new = "Borough")[]}
merged<- merge(nyc311_clean2,nyc_pr_exp, by='Borough',all.x = TRUE)
merged <- merged[,c(1,8,58)]
merged <- distinct(merged)
merged <- na.omit(merged)
head(merged)
```

```
##   Borough                                        Agency Name      exp
## 1   BRONX Department of Housing Preservation and Development 7.390135
## 2   BRONX                       Department of Transportation 7.390135
## 3   BRONX                                         Bronx 05 7.390135
## 4   BRONX                  New York City Police Department 7.390135
## 5   BRONX          Department of Health and Mental Hygiene 7.390135
## 6   BRONX                                        DRIE Unit 7.390135
```

Data Dictionary

1. Borough - This is common column on which the join was performed. It specifies the Borough in which the complaint originated in the nyc311 dataset and it specifies the area in which a particular government job is in the nyc_payroll dataset
2. Agency Name - This is the name of the agency handling the problem.
3. exp - This is the average amount of experience by borough, calculated from the nyc_payroll data and then joined to the nyc311 data on the column 'Borough'

Conclusion

In this assignment, we explored our NYC payroll dataset for better understanding. It helped us in understanding the pay scale of every agency that is operating in NYC. We used those statistics to calculate the average years of experience of people working in each Borough. Finally, we joined our NYC payroll dataset with nyc311 dataset and provided data dictionary for it.