homework iv

Parthivi Shrivastava, Khavya Seshadri

2019-10-03

Introduction

We are tidying the 311 data by removing the infelicities in columns like Incident.Zip , Borough , Closed.Date, etc. and found the unspecified boroughs using the zip code information. Furthermore, we are using the tidyr functions like gather, spread, separate and complete to depict information in the form of tables which can be used for visualization purposes. We are also introducing a relatable dataset, which in our case is NYPD NYC crimes data. We are taking a sample of around 95K from the original dataset which was around 5.5M. We are cleaning this dataset and also using tidyr functions on it.

Initialization

Here we load the tidyverse packages and the `data.table` package and load the nyc311 data set. Then we fix the column names of the nyc311 data so that they have no spaces.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.1     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
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
## Warning: package 'forcats' was built under R version 3.5.2
```

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

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

```
## Warning: package 'data.table' was built under R version 3.5.2
```

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

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

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

```
nyc311<-fread("311_Service_Requests_from_2010_to_Present.csv",
              na.strings = c("","NA","N/A"))
names(nyc311)<-names(nyc311) %>%
  stringr::str_replace_all("\\s", ".")
```

Data pre-processing

Here we perform data pre-processing steps, by dropping irrelevant columns and removing duplicate rows from the nyc311 dataset.

```
nyc311 <- nyc311[,c(-1,-10:-19,-23, -25:-49)]
nyc311nodups <- distinct(nyc311)
names(nyc311nodups)
```

```
##  [1] "Created.Date"                   "Closed.Date"
##  [3] "Agency"                         "Agency.Name"
##  [5] "Complaint.Type"                 "Descriptor"
##  [7] "Location.Type"                  "Incident.Zip"
##  [9] "Status"                         "Due.Date"
## [11] "Resolution.Action.Updated.Date" "Borough"
## [13] "Latitude"                       "Longitude"
## [15] "Location"
```

Handling missing values

In the following snippet, we have handled the missing values and the infelicities in the columns of the data. Intially, we replaced the invalid zip codes with NA.The criteria we used to ensure the validity of the zip code in the data is : 1. Zipcode length should be 5 or 10 . 2. If the zipcode length is 10, then it should satisfy the format of xxxxx-xxxx. Apart from the above rules, we also found zipcodes like 00000, 10000 which were invalid, hence replaced them with NA. Now considering the closed date column, we had dates that were defaulted to 01/01/1900 and also there were around 1 lakh records with closed date lesser than the created date, which seems to be invalid and hence we replaced them with NA. For borough, there were around 8 lakh records with unspecified values, out of which 6 lakh had valid zip codes, so we found the boroughs for those records using the valid zipcode information and remaining we filled with NA.

```r
# Replacing invalid zipcodes with NA
nyc311nodups[Incident.Zip=="00000" | (str_length(str_trim(Incident.Zip))<5 |
        (str_length(str_trim(Incident.Zip)) > 5 &
           str_length(str_trim(Incident.Zip)) < 10)  |
          Incident.Zip=="10000","Incident.Zip"] <- NA

nyc311nodups[as.Date(nyc311nodups$Closed.Date, format="%m/%d/%Y")==
               as.Date("01/01/1900", format="%m/%d/%Y") |
               as.Date(nyc311nodups$Closed.Date, format="%m/%d/%Y")<
                 as.Date(nyc311nodups$Created.Date, format="%m/%d/%Y"),
             c("Closed.Date") ] <- NA

unspecifiedBro <- nyc311nodups %>%
  select(Incident.Zip, Borough) %>%
  filter(Borough=="Unspecified" & !is.na(Incident.Zip))

zipCodeTable <- nyc311nodups %>%
  select(Incident.Zip, Borough) %>%
  filter(Borough!="Unspecified" & (str_length(str_trim(Incident.Zip))==5 |
   (str_length(str_trim(Incident.Zip))==10 & (str_detect(Incident.Zip,'-')))))
zipCodeTable <- distinct(zipCodeTable)
zipCodeTable <-  zipCodeTable %>%
 group_by(Incident.Zip) %>%
 summarize(Borough = first(Borough))

joinedTab <- merge(x=unspecifiedBro, y=zipCodeTable, by = "Incident.Zip", all.x = TRUE)
joinedTab <- distinct(joinedTab)
colnames(joinedTab)[colnames(joinedTab)=="Borough.x"] <- "Borough"

nyc311nodups <- merge(x=nyc311nodups, y=joinedTab,
                 by=c("Incident.Zip", "Borough"), sort=FALSE, all.x = TRUE)
nyc311nodups[!is.na(Borough.y), "Borough"] <- nyc311nodups[!is.na(Borough.y), "Borough.y"]
nyc311nodups[Borough=="Unspecified", "Borough"] <-
  nyc311nodups[Borough=="Unspecified", "Borough.y"]
# drop the borough.y
nyc311nodups <- nyc311nodups[,-"Borough.y"]
head(nyc311nodups)
```

```
##    Incident.Zip   Borough           Created.Date            Closed.Date
## 1:        10465     BRONX 04/14/2015 02:14:40 AM 04/14/2015 03:03:22 AM
## 2:        11234  BROOKLYN 04/14/2015 02:10:12 AM                   <NA>
## 3:        11204  BROOKLYN 04/14/2015 02:03:01 AM                   <NA>
## 4:        11211  BROOKLYN 04/14/2015 02:02:40 AM                   <NA>
## 5:        10025 MANHATTAN 04/14/2015 02:00:04 AM 04/14/2015 02:47:33 AM
## 6:        11205  BROOKLYN 04/14/2015 01:52:15 AM 04/14/2015 02:11:10 AM
##    Agency                      Agency.Name          Complaint.Type
## 1:   NYPD New York City Police Department                 Vending
## 2:   NYPD New York City Police Department        Blocked Driveway
## 3:   NYPD New York City Police Department Noise - Street/Sidewalk
## 4:   NYPD New York City Police Department Noise - Street/Sidewalk
## 5:   NYPD New York City Police Department Noise - Street/Sidewalk
## 6:   NYPD New York City Police Department Noise - Street/Sidewalk
##            Descriptor   Location.Type   Status              Due.Date
```

```
## 1: In Prohibited Area Street/Sidewalk   Closed 04/14/2015 10:14:40 AM
## 2:          No Access Street/Sidewalk     Open 04/14/2015 10:10:12 AM
## 3:   Loud Music/Party Street/Sidewalk     Open 04/14/2015 10:03:01 AM
## 4:       Loud Talking Street/Sidewalk Assigned 04/14/2015 10:02:40 AM
## 5:       Loud Talking Street/Sidewalk   Closed 04/14/2015 10:00:04 AM
## 6:       Loud Talking Street/Sidewalk   Closed 04/14/2015 09:52:15 AM
##    Resolution.Action.Updated.Date Latitude Longitude
## 1:         04/14/2015 03:03:05 AM 40.82573 -73.82111
## 2:                           <NA> 40.61879 -73.93771
## 3:                           <NA> 40.61859 -73.99846
## 4:         04/14/2015 02:10:32 AM 40.71410 -73.95589
## 5:         04/14/2015 02:04:59 AM 40.79792 -73.96385
## 6:         04/14/2015 02:11:10 AM 40.68833 -73.96481
##                                  Location
## 1:   (40.8257259931145, -73.82111429330192)
## 2: (40.618794391821936, -73.93770589155426)
## 3:  (40.61859442131066, -73.99845832101916)
## 4:  (40.71409874640673, -73.95589458206499)
## 5:  (40.79791780509379, -73.96384631347463)
## 6:  (40.68832571866554, -73.96481079590191)
```

Usage of TidyR

In the following snippet, we are showing a table which depicts the frequency of complaints across every borough with respect to every complaint type. We have achieved this by using spread function on the borough column.

```
subsetData <- select(nyc311nodups, Complaint.Type, Borough)
subsetData <- subsetData %>%
    filter(!is.na(Borough)) %>%
  group_by(Complaint.Type,Borough) %>%
  summarize(count=n())
newData <- complete(subsetData, Complaint.Type, Borough)
boroughSpread <- newData %>%
  spread(key=Borough, value=count)
boroughSpread[is.na(boroughSpread)] <- 0
boroughSpread
```

```
## # A tibble: 225 x 6
## # Groups:   Complaint.Type [225]
##    Complaint.Type           BRONX BROOKLYN MANHATTAN QUEENS `STATEN ISLAND`
##    <chr>                    <dbl>    <dbl>     <dbl>  <dbl>           <dbl>
##  1 Adopt-A-Basket              31       51        49     55              11
##  2 Agency Issues                1        0         0      0               0
##  3 Air Quality               3291     9346     16450   5849            1050
##  4 Animal Abuse              3205     3650      1997   3314             957
##  5 Animal Facility - No Pe~    37       95       105    103              39
##  6 Animal in a Park          1091     1902      3137   1564             518
##  7 APPLIANCE                16113    17694      8813   6817            1243
##  8 Asbestos                  1255     2863      3796   2049             349
##  9 Beach/Pool/Sauna Compla~    80      215       202    238             161
## 10 Benefit Card Replacement     0        0         1      0               0
```

```
## # ... with 215 more rows
```

In the following snippet, we are showing a table which depicts the frequency of complaints for the top 5 agencies with respect to every complaint type. We have achieved this by using group by function which is similar to gather in tidyr library.

```
AgencyCount <- select(nyc311nodups,Complaint.Type , Agency)
(agencyData <- AgencyCount %>%
  filter(Agency== "HPD" | Agency == "DOT" | Agency=="NYPD"
         | Agency == "DEP" | Agency=="DSNY") %>%
  group_by(Complaint.Type,Agency) %>%
  summarize(frequency = n()))
```

```
## # A tibble: 143 x 3
## # Groups:   Complaint.Type [129]
##    Complaint.Type   Agency frequency
##    <chr>            <chr>      <int>
##  1 Adopt-A-Basket   DSNY         197
##  2 Agency Issues    DEP            1
##  3 Agency Issues    DOT          553
##  4 Agency Issues    DSNY         920
##  5 Agency Issues    NYPD           2
##  6 Air Quality      DEP        36034
##  7 Animal Abuse     NYPD       13126
##  8 Animal in a Park DEP            5
##  9 APPLIANCE        HPD        50690
## 10 Asbestos         DEP         7584
## # ... with 133 more rows
```

In the following snippet, we are showing a table which depicts the year wise frequency of complaints with respect to every borough. We have achieved this by using separate function to extract the year from the created date, after which we spreaded across the year, thus computing the frequency of complaints for each borough.

```
boroughYear <-nyc311nodups %>%
  select( Borough , Created.Date, Complaint.Type) %>%
  filter(!is.na(Borough))
yearData <- separate(boroughYear, Created.Date, into=c("month", "day", "year"),
                     convert = T)
```

```
## Warning: Expected 3 pieces. Additional pieces discarded in 8012461 rows [1,
## 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
boroughYear <- yearData %>%
  group_by(year, Borough) %>%
  summarize(frequency=n())
(yearSpread <- boroughYear %>%
  spread(key=year, value=frequency))
```

```
## # A tibble: 5 x 14
##   Borough `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010` `2011`
```

```
##   <chr>    <int>  <int>  <int>  <int>  <int>  <int>  <int>  <int>  <int>
## 1 BRONX     1907    808      7    374    434    631   3198 294858 275932
## 2 BROOKL~   5391   2186     63    839    942   1219   5188 490283 465870
## 3 MANHAT~   6911   2744    393   1239   1251   1744   5755 315889 298611
## 4 QUEENS    5336   2314     47    696    792   1327   4331 389379 355607
## 5 STATEN~   2015    761      2   1373   1621   1855   2432  85656  85533
## # ... with 4 more variables: `2012` <int>, `2013` <int>, `2014` <int>,
## #   `2015` <int>
```

In the following snippet, we are showing a table which depicts the frequency of complaints across every borough with respect to the top 5 agencies. We have achieved this using spread function on the borough column.

```
AgencyBorough <- select(nyc311nodups,Agency, Borough)
AgencyBorough <- AgencyBorough %>%
  filter((Agency== "HPD" | Agency == "DOT" | Agency=="NYPD"
          | Agency == "DEP" | Agency=="DSNY")
         & !is.na(Borough)) %>%
  group_by( Agency,Borough) %>%
  summarize(count= n())
(AgencyBorough <- AgencyBorough %>%
  spread(key = Borough, value = count))
```

```
## # A tibble: 5 x 6
## # Groups:   Agency [5]
##   Agency  BRONX BROOKLYN MANHATTAN QUEENS `STATEN ISLAND`
##   <chr>   <int>    <int>     <int>  <int>           <int>
## 1 DEP    102219   220068    208776 235471           69578
## 2 DOT    216610   446116    325233 443452          121491
## 3 DSNY    90736   234527     84728 209729           62443
## 4 HPD    748894   881767    471602 321098           44890
## 5 NYPD   130312   330500    223584 291644           46410
```

Relatable data set - NYPD NYC Crimes data

Description

We have used the NYPD NYC crimes data which is a sample of size 95,593 records taken from the original data source.This dataset includes all valid felony, misdemeanor, and violation crimes reported to the New York City Police Department (NYPD).

Initialization

Here we load the NYC Crimes data set from the link as provided below and we fill the empty cells with NA.

```
nycCrimes <-
  fread("https://raw.githubusercontent.com/jamesjynus/Shiny/master/data/crime.csv",
                  na.strings = c("","NA"))
```

Data pre-processing of NYC Crimes data

Here, we removed the irrelevant columns and duplicate records in the data, fixed the column name for borough and we are showing the head and data dictionary.

```
nycCrimes <- nycCrimes[,c(-1,-2,-10,-13,-14,-15,-17)]
nycCrimenodups <- distinct(nycCrimes)
colnames(nycCrimenodups)[colnames(nycCrimenodups)=="Boro"] <- "Borough"
nycCrimenodups <-  nycCrimenodups[str_trim(Offense)!="",]
names(nycCrimenodups)
```

```
##  [1] "Date"           "Time"           "Code"           "Offense"
##  [5] "Status"         "Type"           "Borough"        "Latitude"
##  [9] "Longitude"      "Population"     "Year_Month_New"
```

```
head(nycCrimenodups)
```

```
##          Date     Time Code                         Offense    Status
## 1: 2006-03-10 14:30:00  113                         FORGERY COMPLETED
## 2: 2012-12-19 10:00:00  344   ASSAULT 3 & RELATED OFFENSES COMPLETED
## 3: 2011-10-14 14:20:00  126        MISCELLANEOUS PENAL LAW COMPLETED
## 4: 2009-07-31 11:50:00  109                  GRAND LARCENY ATTEMPTED
## 5: 2006-01-23 17:45:00  341                  PETIT LARCENY COMPLETED
## 6: 2013-09-09 21:47:00  359 OFFENSES AGAINST PUBLIC ADMINI COMPLETED
##           Type       Borough Latitude Longitude Population Year_Month_New
## 1:      FELONY      BROOKLYN 40.66200 -73.91959    2465690        2006-03
## 2: MISDEMEANOR STATEN ISLAND 40.57112 -74.09007     471000        2012-12
## 3:      FELONY     MANHATTAN 40.79967 -73.94720    1595517        2011-10
## 4:      FELONY        QUEENS 40.76480 -73.77161    2230000        2009-07
## 5: MISDEMEANOR     MANHATTAN 40.77365 -73.95986    1566766        2006-01
## 6: MISDEMEANOR         BRONX 40.81937 -73.91828    1420414        2013-09
```

Data Dictionary

- Date - Date on which crime happened in the format yyyy-mm-dd.
- Time - Time at which crime occured in the format hh:mm:ss.
- Code - Unique code for every offense.
- Offense - The description of the crime type(sub-categories of the crime).
- Status - The status of the crime report submitted(Allowed values: COMPLETED , ATTEMPTED).
- Type - The type of crime(Allowed types: FELONY, MISDEMEANOR,VIOLATION).
- Borough - town/district of the NYC provided by submitter(Values: BRONX, BROOKLYN, MANHATTAN, QUEENS, STATEN ISLAND).
- Latitude - Geo-based latitude of the incident location(Type: degrees).
- Longitude – Geo-based longitude of the incident location(Type: degrees).
- Population - The population of the Borough on the date of the crime.
- Year_Month_New - Year and Month of the crime date in the format yyyy-mm.

Usage of TidyR

In the following snippet, we are showing a table which depicts the frequency of crimes across every borough with respect to every crime type. We have achieved this by using spread function on the borough column.

```
subsetData <- select(nycCrimenodups, Type, Borough)
subsetData <- subsetData %>%
    filter(!is.na(Borough)) %>%
  group_by(Type,Borough) %>%
  summarize(count=n())
newData <- complete(subsetData,Type, Borough)
boroughSpread <- newData %>%
  spread(key=Borough, value=count)
boroughSpread[is.na(boroughSpread)] <- 0
boroughSpread
```

```
## # A tibble: 3 x 6
## # Groups:   Type [3]
##   Type        BRONX BROOKLYN MANHATTAN QUEENS `STATEN ISLAND`
##   <chr>       <int>    <int>     <int>  <int>           <int>
## 1 FELONY       5573     9216      7379   6341             955
## 2 MISDEMEANOR 12508    15780     13253   9724            2641
## 3 VIOLATION    2549     3647      2301   2477             883
```

In the following snippet, we are showing a table which depicts the year wise frequency of crimes for each borough. We have achieved this by using the separate function to extract the year from the created date, and then we spread across the year, thus computing the frequency of crimes for each borough.

```
boroYear <-nycCrimenodups %>%
  select( Borough , Year_Month_New,Type) %>%
  filter(!is.na(Borough))
yearData <- separate(boroYear, Year_Month_New, into=c("year", "month"), convert = T)
boroYear <- yearData %>%
  group_by(year,Borough) %>%
  summarize(frequency=n())
(yearSpread <- boroYear %>%
  spread(key=year, value=frequency))
```

```
## # A tibble: 5 x 12
##   Borough `2006` `2007` `2008` `2009` `2010` `2011` `2012` `2013` `2014`
##   <chr>    <int>  <int>  <int>  <int>  <int>  <int>  <int>  <int>  <int>
## 1 BRONX     1832   2004   1950   1928   1967   1792   1812   1830   1836
## 2 BROOKL~   2641   2672   2688   2619   2658   2687   2626   2597   2573
## 3 MANHAT~   2203   2204   2244   2223   2035   1977   2013   1980   1996
## 4 QUEENS    1786   1772   1778   1608   1624   1652   1654   1635   1698
## 5 STATEN~    458    488    458    434    376    384    376    373    386
## # ... with 2 more variables: `2015` <int>, `2016` <int>
```

Conclusion

In this document, we introduced a new dataset: NYPD NYC Crimes data relateable to our 311NYC data. We performed data cleaning by dropping the irrelevant columns, removing the duplicates and replacing

the missing values on both the datasets. We have also made use of the tidyR functions, showing relevant information with respect to complaints and crimes in the form of tables.