assignment\_5

Priya Shaji

2/28/2019

## Tidying and Transforming Data

### loading the required packages

#install.packages("tidyr")  
#install.packages("dplyr")  
library(tidyr)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

### Dataset

#### Create a dataframe and load the dataset from the .csv file

df\_flight\_wide <- read.csv("/Users/priyashaji/Documents/cuny msds/Spring'19/data 607/assignments/assignment\_5/FlightsInfo - Sheet1.csv",header=TRUE,stringsAsFactors = FALSE)

## Warning in read.table(file = file, header = header, sep = sep, quote  
## = quote, : incomplete final line found by readTableHeader on '/Users/  
## priyashaji/Documents/cuny msds/Spring'19/data 607/assignments/assignment\_5/  
## FlightsInfo - Sheet1.csv'

df\_flight\_wide

## X X.1 Los.Angeles Phoenix San.Diego San.Franciso Seattle  
## 1 ALASKA on time 497 221 212 503 1,841  
## 2 delayed 62 12 20 102 305  
## 3 AM WEST on time 694 4,840 383 320 201  
## 4 delayed 117 415 65 129 61

### Tidying and Transforming data

Using functions provided by tidyr and dplyr packages The current dataframe: df\_flight is a wide format of data We will use gather() function of tidyr package to make the data in long format

gather() function uses the folowing arguments:

data: Data object

key: Name of new key column (made from names of data columns)

value: Name of new value column

Names of source columns that contain values

factor\_key: Treat the new key column as a factor (instead of character vector)

df\_flight\_long <- gather(df\_flight\_wide, city, number, Los.Angeles:Seattle, factor\_key=TRUE)  
df\_flight\_long

## X X.1 city number  
## 1 ALASKA on time Los.Angeles 497  
## 2 delayed Los.Angeles 62  
## 3 AM WEST on time Los.Angeles 694  
## 4 delayed Los.Angeles 117  
## 5 ALASKA on time Phoenix 221  
## 6 delayed Phoenix 12  
## 7 AM WEST on time Phoenix 4,840  
## 8 delayed Phoenix 415  
## 9 ALASKA on time San.Diego 212  
## 10 delayed San.Diego 20  
## 11 AM WEST on time San.Diego 383  
## 12 delayed San.Diego 65  
## 13 ALASKA on time San.Franciso 503  
## 14 delayed San.Franciso 102  
## 15 AM WEST on time San.Franciso 320  
## 16 delayed San.Franciso 129  
## 17 ALASKA on time Seattle 1,841  
## 18 delayed Seattle 305  
## 19 AM WEST on time Seattle 201  
## 20 delayed Seattle 61

rename the columns

colnames(df\_flight\_long)[2] <- "Status"   
colnames(df\_flight\_long)[1] <- "Airline"  
df\_flight\_long$Airline <- factor(df\_flight\_long$Airline)  
df\_flight\_long

## Airline Status city number  
## 1 ALASKA on time Los.Angeles 497  
## 2 delayed Los.Angeles 62  
## 3 AM WEST on time Los.Angeles 694  
## 4 delayed Los.Angeles 117  
## 5 ALASKA on time Phoenix 221  
## 6 delayed Phoenix 12  
## 7 AM WEST on time Phoenix 4,840  
## 8 delayed Phoenix 415  
## 9 ALASKA on time San.Diego 212  
## 10 delayed San.Diego 20  
## 11 AM WEST on time San.Diego 383  
## 12 delayed San.Diego 65  
## 13 ALASKA on time San.Franciso 503  
## 14 delayed San.Franciso 102  
## 15 AM WEST on time San.Franciso 320  
## 16 delayed San.Franciso 129  
## 17 ALASKA on time Seattle 1,841  
## 18 delayed Seattle 305  
## 19 AM WEST on time Seattle 201  
## 20 delayed Seattle 61

As we see in the above dataframe, “Airline” has alternate missing values. In order to fill the missing values from the Airline name previous to it, we use na.locf(.) function. Therefore, we first install the package “zoo”

na.locf(.) :

Last Observation Carried Forward Generic function for replacing each NA with the most recent non-NA prior to it.

#install.packages("zoo")  
library(zoo)

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

df\_flight\_long[df\_flight\_long$Airline=="","Airline"] <- NA  
df\_flight\_long <- df\_flight\_long %>%  
 do(na.locf(.))  
df\_flight\_long

## Airline Status city number  
## 1 ALASKA on time Los.Angeles 497  
## 2 ALASKA delayed Los.Angeles 62  
## 3 AM WEST on time Los.Angeles 694  
## 4 AM WEST delayed Los.Angeles 117  
## 5 ALASKA on time Phoenix 221  
## 6 ALASKA delayed Phoenix 12  
## 7 AM WEST on time Phoenix 4,840  
## 8 AM WEST delayed Phoenix 415  
## 9 ALASKA on time San.Diego 212  
## 10 ALASKA delayed San.Diego 20  
## 11 AM WEST on time San.Diego 383  
## 12 AM WEST delayed San.Diego 65  
## 13 ALASKA on time San.Franciso 503  
## 14 ALASKA delayed San.Franciso 102  
## 15 AM WEST on time San.Franciso 320  
## 16 AM WEST delayed San.Franciso 129  
## 17 ALASKA on time Seattle 1,841  
## 18 ALASKA delayed Seattle 305  
## 19 AM WEST on time Seattle 201  
## 20 AM WEST delayed Seattle 61

# convert the column "number" from type "character" to type "numeric"  
df\_flight\_long$number<-suppressWarnings(as.numeric(df\_flight\_long$number))  
is.numeric(df\_flight\_long$number)

## [1] TRUE

Our dataframe is in long format now, we can apply dataset

We will use the spread() function.

spread() returns a copy of your data set that has had the key and value columns removed. In their place, spread() adds a new column for each unique value of the key column. These unique values will form the column names of the new columns. spread() distributes the cells of the former value column across the cells of the new columns and truncates any non-key, non-value columns in a way that prevents duplication.

df\_flight <- spread(df\_flight\_long, "Status", "number")   
  
is.factor(df\_flight$Airline)

## [1] TRUE

is.numeric(df\_flight$`on time`)

## [1] TRUE

is.numeric(df\_flight$delayed)

## [1] TRUE

is.factor(df\_flight$city)

## [1] TRUE

df\_flight

## Airline city delayed on time  
## 1 ALASKA Los.Angeles 62 497  
## 2 ALASKA Phoenix 12 221  
## 3 ALASKA San.Diego 20 212  
## 4 ALASKA San.Franciso 102 503  
## 5 ALASKA Seattle 305 NA  
## 6 AM WEST Los.Angeles 117 694  
## 7 AM WEST Phoenix 415 NA  
## 8 AM WEST San.Diego 65 383  
## 9 AM WEST San.Franciso 129 320  
## 10 AM WEST Seattle 61 201

### Analysis of flight status

Analysis to compare the arrival delays for the two airlines

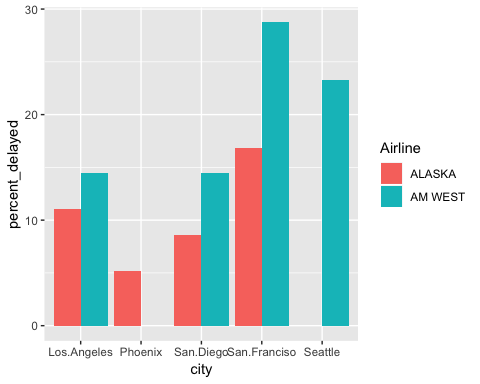
Using the pipe(%>%) operator to do the analysis

delay\_analysis<- as.data.frame(df\_flight %>%  
 mutate(total = delayed + `on time`) %>%  
 group\_by(Airline,city) %>%  
 summarise(delayed\_total = sum(delayed),   
 on\_time\_total = sum(`on time`),  
 count\_total = sum(total),  
 percent\_ontime = round(on\_time\_total/count\_total\*100,digit = 2),  
 percent\_delayed = round(delayed\_total/count\_total\*100,digit = 2) ))  
delay\_analysis

## Airline city delayed\_total on\_time\_total count\_total  
## 1 ALASKA Los.Angeles 62 497 559  
## 2 ALASKA Phoenix 12 221 233  
## 3 ALASKA San.Diego 20 212 232  
## 4 ALASKA San.Franciso 102 503 605  
## 5 ALASKA Seattle 305 NA NA  
## 6 AM WEST Los.Angeles 117 694 811  
## 7 AM WEST Phoenix 415 NA NA  
## 8 AM WEST San.Diego 65 383 448  
## 9 AM WEST San.Franciso 129 320 449  
## 10 AM WEST Seattle 61 201 262  
## percent\_ontime percent\_delayed  
## 1 88.91 11.09  
## 2 94.85 5.15  
## 3 91.38 8.62  
## 4 83.14 16.86  
## 5 NA NA  
## 6 85.57 14.43  
## 7 NA NA  
## 8 85.49 14.51  
## 9 71.27 28.73  
## 10 76.72 23.28

library(ggplot2)  
delays <- ggplot(delay\_analysis,aes(x = city, y = percent\_delayed))  
suppressWarnings(delays + geom\_bar(stat = "identity", position="dodge", aes(fill = Airline)))

## Warning: Removed 2 rows containing missing values (geom\_bar).



#### Conclusion

As we can see by the graph, the delayed rate of AM WEST is higher than ALASKA Airline. The delayed rate for AM WEST is heighest for Phoenix.