Project\_2

Priya Shaji

3/5/2019

Table of Contents

## Preparing different datasets for downstream analysis work

### 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

### Datasets

#### Dataset of relation between income and religion in U.S experiment

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

This dataset explores the relationship between income and religion in the US. It comes from a report produced by the Pew Research Center, an American think-tank that collects data on attitudes to topics ranging from religion to the internet, and produces many reports that contain datasets in this format.

library(tibble)  
d1 <- as\_tibble(read.csv("/Users/priyashaji/Documents/cuny msds/Spring'19/data 607/projects/project\_2/thinktank .csv", stringsAsFactors = FALSE, check.names = FALSE))  
d1

## # A tibble: 18 x 11  
## religion `<$10k` `$10-20k` `$20-30k` `$30-40k` `$40-50k` `$50-75k`  
## <chr> <int> <int> <int> <int> <int> <int>  
## 1 Agnostic 27 34 60 81 76 137  
## 2 Atheist 12 27 37 52 35 70  
## 3 Buddhist 27 21 30 34 33 58  
## 4 Catholic 418 617 732 670 638 1116  
## 5 Don’t k… 15 14 15 11 10 35  
## 6 Evangel… 575 869 1064 982 881 1486  
## 7 Hindu 1 9 7 9 11 34  
## 8 Histori… 228 244 236 238 197 223  
## 9 Jehovah… 20 27 24 24 21 30  
## 10 Jewish 19 19 25 25 30 95  
## 11 Mainlin… 289 495 619 655 651 1107  
## 12 Mormon 29 40 48 51 56 112  
## 13 Muslim 6 7 9 10 9 23  
## 14 Orthodox 13 17 23 32 32 47  
## 15 Other C… 9 7 11 13 13 14  
## 16 Other F… 20 33 40 46 49 63  
## 17 Other W… 5 2 3 4 2 7  
## 18 Unaffil… 217 299 374 365 341 528  
## # … with 4 more variables: `$75-100k` <int>, `$100-150k` <int>,  
## # `>150k` <int>, `Don't know/refused` <int>

##### Tidying and Transforming Data

This dataset has three variables, religion, income and frequency. To tidy it, we need to gather the non-variable columns into a two-column key-value pair. This action is often described as making a wide dataset long (or tall), but I’ll avoid those terms because they’re imprecise.

When gathering variables, we need to provide the name of the new key-value columns to create. The first argument, is the name of the key column, which is the name of the variable defined by the values of the column headings. In this case, it’s income. The second argument is the name of the value column, frequency. The third argument defines the columns to gather, here, every column except religion.

d1\_clean<-d1 %>%  
 gather(income, frequency, -religion)

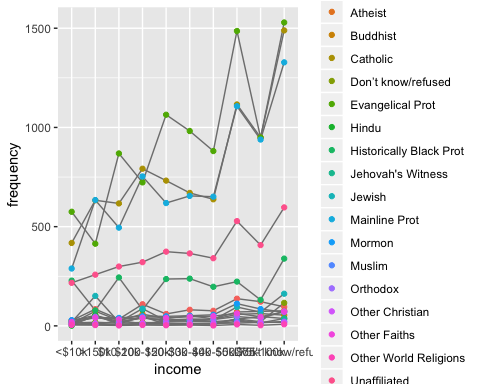
d1\_clean

## # A tibble: 180 x 3  
## religion income frequency  
## <chr> <chr> <int>  
## 1 Agnostic <$10k 27  
## 2 Atheist <$10k 12  
## 3 Buddhist <$10k 27  
## 4 Catholic <$10k 418  
## 5 Don’t know/refused <$10k 15  
## 6 Evangelical Prot <$10k 575  
## 7 Hindu <$10k 1  
## 8 Historically Black Prot <$10k 228  
## 9 Jehovah's Witness <$10k 20  
## 10 Jewish <$10k 19  
## # … with 170 more rows

This form is tidy because each column represents a variable and each row represents an observation, in this case a demographic unit corresponding to a combination of religion and income.

Plot a graph between income and frequency which is grouped by religion

library(ggplot2)  
ggplot(d1\_clean, aes(income, frequency)) +   
 geom\_line(aes(group = religion), colour = "grey50") +   
 geom\_point(aes(colour = religion))



In graph we see, the highest no. of people which has a salary range below 10k $ belong to evangelical prot

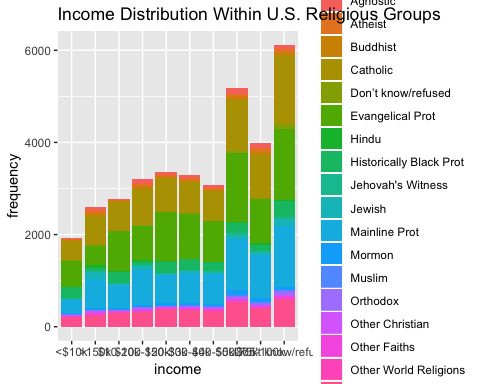
And the highest no. of people which has a salary range between >150k $ belong to Mainline Prot followed by catholic group.

Calculate the mean for the number of people in each religion group using the aggregate function. There are total 18 groups of religion.

aggregate(frequency ~ religion , d1\_clean, mean)

## religion frequency  
## 1 Agnostic 82.6  
## 2 Atheist 51.5  
## 3 Buddhist 41.1  
## 4 Catholic 805.4  
## 5 Don’t know/refused 27.2  
## 6 Evangelical Prot 947.2  
## 7 Hindu 25.7  
## 8 Historically Black Prot 199.5  
## 9 Jehovah's Witness 21.5  
## 10 Jewish 68.2  
## 11 Mainline Prot 747.0  
## 12 Mormon 58.1  
## 13 Muslim 11.6  
## 14 Orthodox 36.3  
## 15 Other Christian 12.9  
## 16 Other Faiths 44.9  
## 17 Other World Religions 4.2  
## 18 Unaffiliated 370.7

ggplot(data=d1\_clean, aes(x=income, y=frequency, fill=religion)) + geom\_bar(stat="identity",position = 'stack') + ggtitle('Income Distribution Within U.S. Religious Groups')



##### Conclusion

From the graph we can see that religious tradition clearly varies by income level. We can see that for the highest income category (>$150k), ‘Mainline Prot’ followed by ‘ Catholic’ have the highest proportions. If we examine the lowest income category (<$10k) we see that ‘Evangelical Prot’ have the highest proportions

#### Dataset of Engilsh Premier League

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

#Tranform CSV.file into a tbl\_df so it prints tables in a more friendly way.  
leaguedata<- read.csv(file="https://raw.githubusercontent.com/yli74/movies/master/Engilsh%20Premier%20League%20Data.csv",stringsAsFactors = FALSE,sep = ",")  
leaguedata\_tbl=tbl\_df(leaguedata)  
leaguedata\_tbl

## # A tibble: 20 x 20  
## Team P W D L GF GA GD Pts PPG Wh Dh  
## <chr> <int> <int> <int> <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 Manch… 7 6 0 1 18 7 11 18 2.57 3 0  
## 2 Totte… 7 5 2 0 12 3 9 17 2.43 3 1  
## 3 Arsen… 7 5 1 1 16 7 9 16 2.29 2 0  
## 4 Liver… 7 5 1 1 18 10 8 16 2.29 2 0  
## 5 Evert… 7 4 2 1 11 5 6 14 2 2 2  
## 6 Manch… 7 4 1 2 13 8 5 13 1.86 2 1  
## 7 Chels… 7 4 1 2 12 9 3 13 1.86 2 0  
## 8 Cryst… 7 3 2 2 11 8 3 11 1.57 1 1  
## 9 West … 7 2 3 2 8 7 1 9 1.29 1 1  
## 10 South… 7 2 3 2 7 6 1 9 1.29 1 2  
## 11 Watfo… 7 2 2 3 12 13 -1 8 1.14 1 1  
## 12 Leice… 7 2 2 3 8 11 -3 8 1.14 2 2  
## 13 Bourn… 7 2 2 3 6 11 -5 8 1.14 2 0  
## 14 Burnl… 7 2 1 4 5 9 -4 7 1 2 1  
## 15 Hull … 7 2 1 4 7 14 -7 7 1 1 0  
## 16 Middl… 7 1 3 3 7 10 -3 6 0.86 0 1  
## 17 Swans… 7 1 1 5 6 12 -6 4 0.570 0 1  
## 18 West … 7 1 1 5 8 17 -9 4 0.570 1 1  
## 19 Stoke… 7 0 3 4 5 16 -11 3 0.43 0 1  
## 20 Sunde… 7 0 2 5 6 13 -7 2 0.290 0 1  
## # … with 8 more variables: Lh <int>, GFh <int>, GAh <int>, Wa <int>,  
## # Da <int>, La <int>, GFa <int>, Gaa <int>

##### Tidying and Transforming Data

# Description of the Data  
str(leaguedata)

## 'data.frame': 20 obs. of 20 variables:  
## $ Team: chr "Manchester City" "Tottenham" "Arsenal" "Liverpool" ...  
## $ P : int 7 7 7 7 7 7 7 7 7 7 ...  
## $ W : int 6 5 5 5 4 4 4 3 2 2 ...  
## $ D : int 0 2 1 1 2 1 1 2 3 3 ...  
## $ L : int 1 0 1 1 1 2 2 2 2 2 ...  
## $ GF : int 18 12 16 18 11 13 12 11 8 7 ...  
## $ GA : int 7 3 7 10 5 8 9 8 7 6 ...  
## $ GD : int 11 9 9 8 6 5 3 3 1 1 ...  
## $ Pts : int 18 17 16 16 14 13 13 11 9 9 ...  
## $ PPG : num 2.57 2.43 2.29 2.29 2 1.86 1.86 1.57 1.29 1.29 ...  
## $ Wh : int 3 3 2 2 2 2 2 1 1 1 ...  
## $ Dh : int 0 1 0 0 2 1 0 1 1 2 ...  
## $ Lh : int 0 0 1 0 0 1 1 1 1 0 ...  
## $ GFh : int 9 5 8 9 6 8 6 5 5 3 ...  
## $ GAh : int 2 1 5 2 3 4 3 3 4 2 ...  
## $ Wa : int 3 2 3 3 2 2 2 1 1 1 ...  
## $ Da : int 0 1 1 1 0 0 1 2 2 1 ...  
## $ La : int 1 0 0 1 1 1 1 1 1 2 ...  
## $ GFa : int 9 7 8 9 5 5 6 6 3 4 ...  
## $ Gaa : int 5 2 2 8 2 4 6 5 3 4 ...

names(leaguedata)

## [1] "Team" "P" "W" "D" "L" "GF" "GA" "GD" "Pts" "PPG"   
## [11] "Wh" "Dh" "Lh" "GFh" "GAh" "Wa" "Da" "La" "GFa" "Gaa"

Using the functions provided by tidyr and dplyr packages

Select the appropriate columns that are needed. Then calculate the total goals by adding total\_goals\_scored\_home and total\_goals\_scored\_away And also calculating Total\_goals\_conceded\_home by adding goals\_conceded\_at\_home and goals\_conceded\_away

rename those columns for a meaningful look

tidy\_leaguedata <-leaguedata\_tbl %>%  
 select(Team,GF:Pts,-GD,GFh,GAh,GFa,Gaa) %>%  
 mutate(Total\_goals=GF+GA,Total\_goals\_conceded\_home=GAh+Gaa) %>%  
 rename(team=Team,total\_goals\_scored\_home=GF,total\_goals\_scored\_away=GA,goals\_scored\_at\_home=GFh,goals\_scored\_away=GFa,goals\_conceded\_at\_home=GAh,goals\_conceded\_away=Gaa)  
tidy\_leaguedata

## # A tibble: 20 x 10  
## team total\_goals\_sco… total\_goals\_sco… Pts goals\_scored\_at…  
## <chr> <int> <int> <int> <int>  
## 1 Manc… 18 7 18 9  
## 2 Tott… 12 3 17 5  
## 3 Arse… 16 7 16 8  
## 4 Live… 18 10 16 9  
## 5 Ever… 11 5 14 6  
## 6 Manc… 13 8 13 8  
## 7 Chel… 12 9 13 6  
## 8 Crys… 11 8 11 5  
## 9 West… 8 7 9 5  
## 10 Sout… 7 6 9 3  
## 11 Watf… 12 13 8 7  
## 12 Leic… 8 11 8 5  
## 13 Bour… 6 11 8 3  
## 14 Burn… 5 9 7 5  
## 15 Hull… 7 14 7 3  
## 16 Midd… 7 10 6 3  
## 17 Swan… 6 12 4 4  
## 18 West… 8 17 4 4  
## 19 Stok… 5 16 3 2  
## 20 Sund… 6 13 2 4  
## # … with 5 more variables: goals\_conceded\_at\_home <int>,  
## # goals\_scored\_away <int>, goals\_conceded\_away <int>, Total\_goals <int>,  
## # Total\_goals\_conceded\_home <int>

According to the chart found on www.soccerstats.com, home advantage should show the following data, total points, total points scored at home,total goals scored at home

To count the total points scored at home, home advantage 1=(total\_goals\_scored\_home/Total\_goals)\*100)).

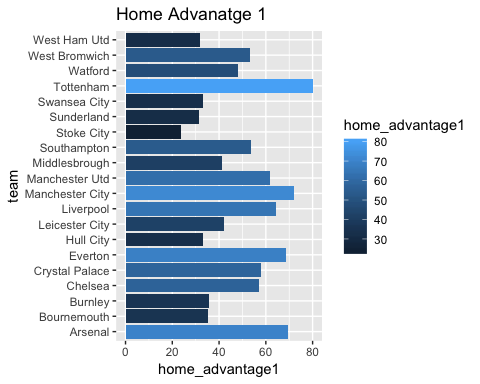
To count the total goals scored at home,home advantage 2 =((goals\_scored\_at\_home/(goals\_scored\_away+goals\_scored\_at\_home)))\*100)

Home\_advantage<-tidy\_leaguedata %>%  
   
 group\_by(team) %>%  
   
 mutate(home\_advantage1=((total\_goals\_scored\_home/Total\_goals)\*100))%>%  
   
 mutate(home\_advantage2=((goals\_scored\_at\_home/(goals\_scored\_away+goals\_scored\_at\_home)))\*100)%>%  
   
 select(team,home\_advantage1,home\_advantage2,Total\_goals)%>%  
   
 arrange(home\_advantage1)  
  
Home\_advantage

## # A tibble: 20 x 4  
## # Groups: team [20]  
## team home\_advantage1 home\_advantage2 Total\_goals  
## <chr> <dbl> <dbl> <int>  
## 1 Stoke City 23.8 40 21  
## 2 Sunderland 31.6 66.7 19  
## 3 West Ham Utd 32 50 25  
## 4 Hull City 33.3 42.9 21  
## 5 Swansea City 33.3 66.7 18  
## 6 Bournemouth 35.3 50 17  
## 7 Burnley 35.7 100 14  
## 8 Middlesbrough 41.2 42.9 17  
## 9 Leicester City 42.1 62.5 19  
## 10 Watford 48 58.3 25  
## 11 West Bromwich 53.3 62.5 15  
## 12 Southampton 53.8 42.9 13  
## 13 Chelsea 57.1 50 21  
## 14 Crystal Palace 57.9 45.5 19  
## 15 Manchester Utd 61.9 61.5 21  
## 16 Liverpool 64.3 50 28  
## 17 Everton 68.8 54.5 16  
## 18 Arsenal 69.6 50 23  
## 19 Manchester City 72 50 25  
## 20 Tottenham 80 41.7 15

To see which team scored the most POINTS at home

library(ggplot2)  
  
ggplot(data=Home\_advantage, aes(x = team, y = home\_advantage1,fill=home\_advantage1)) + geom\_bar(stat="identity",position = 'dodge') + ggtitle('Home Advanatge 1')+coord\_flip()



Tottenham scored the most points at home, Stoke city scored the least

Calculate the mean for the number of points scored by each team group using the aggregate function. There are total 20 teams.

aggregate(home\_advantage1 ~ team , Home\_advantage, mean)

## team home\_advantage1  
## 1 Arsenal 69.56522  
## 2 Bournemouth 35.29412  
## 3 Burnley 35.71429  
## 4 Chelsea 57.14286  
## 5 Crystal Palace 57.89474  
## 6 Everton 68.75000  
## 7 Hull City 33.33333  
## 8 Leicester City 42.10526  
## 9 Liverpool 64.28571  
## 10 Manchester City 72.00000  
## 11 Manchester Utd 61.90476  
## 12 Middlesbrough 41.17647  
## 13 Southampton 53.84615  
## 14 Stoke City 23.80952  
## 15 Sunderland 31.57895  
## 16 Swansea City 33.33333  
## 17 Tottenham 80.00000  
## 18 Watford 48.00000  
## 19 West Bromwich 53.33333  
## 20 West Ham Utd 32.00000

By calculating overall mean, highest mean points are scored by Tottenham, and lowest meanpoints are scored by Stoke City.

Now we will summarize the dataset by grouping it by team and calculating the point\_rate

by\_team\_point <- group\_by(Home\_advantage, team)   
summarize(by\_team\_point, points\_rate<-sum(home\_advantage1)/sum(Total\_goals))

## # A tibble: 20 x 2  
## team `points\_rate <- sum(home\_advantage1)/sum(Total\_goals)`  
## <chr> <dbl>  
## 1 Arsenal 3.02  
## 2 Bournemouth 2.08  
## 3 Burnley 2.55  
## 4 Chelsea 2.72  
## 5 Crystal Palace 3.05  
## 6 Everton 4.30  
## 7 Hull City 1.59  
## 8 Leicester City 2.22  
## 9 Liverpool 2.30  
## 10 Manchester City 2.88  
## 11 Manchester Utd 2.95  
## 12 Middlesbrough 2.42  
## 13 Southampton 4.14  
## 14 Stoke City 1.13  
## 15 Sunderland 1.66  
## 16 Swansea City 1.85  
## 17 Tottenham 5.33  
## 18 Watford 1.92  
## 19 West Bromwich 3.56  
## 20 West Ham Utd 1.28

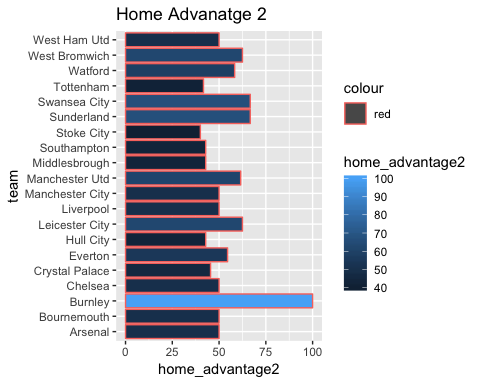
summarize(by\_team\_point, sum(home\_advantage1))

## # A tibble: 20 x 2  
## team `sum(home\_advantage1)`  
## <chr> <dbl>  
## 1 Arsenal 69.6  
## 2 Bournemouth 35.3  
## 3 Burnley 35.7  
## 4 Chelsea 57.1  
## 5 Crystal Palace 57.9  
## 6 Everton 68.8  
## 7 Hull City 33.3  
## 8 Leicester City 42.1  
## 9 Liverpool 64.3  
## 10 Manchester City 72   
## 11 Manchester Utd 61.9  
## 12 Middlesbrough 41.2  
## 13 Southampton 53.8  
## 14 Stoke City 23.8  
## 15 Sunderland 31.6  
## 16 Swansea City 33.3  
## 17 Tottenham 80   
## 18 Watford 48   
## 19 West Bromwich 53.3  
## 20 West Ham Utd 32

By calculating overall point\_rate, highest mean points are scored by Tottenham, and lowest meanpoints are scored by Stoke City.

To see which team scored the most GOALS at home

ggplot(data=Home\_advantage, aes(x = team, y = home\_advantage2,fill=home\_advantage2,color="red")) + geom\_bar(stat="identity",position = 'dodge') + ggtitle('Home Advanatge 2')+coord\_flip()



Burnley scored the most goals at home, Stoke city scored the least.

Now we will summarize the dataset by grouping it by team and calculating the goal\_rate

by\_team\_goal <- group\_by(Home\_advantage, team)   
summarize(by\_team\_goal, goal\_rate<-sum(home\_advantage2)/sum(Total\_goals))

## # A tibble: 20 x 2  
## team `goal\_rate <- sum(home\_advantage2)/sum(Total\_goals)`  
## <chr> <dbl>  
## 1 Arsenal 2.17  
## 2 Bournemouth 2.94  
## 3 Burnley 7.14  
## 4 Chelsea 2.38  
## 5 Crystal Palace 2.39  
## 6 Everton 3.41  
## 7 Hull City 2.04  
## 8 Leicester City 3.29  
## 9 Liverpool 1.79  
## 10 Manchester City 2   
## 11 Manchester Utd 2.93  
## 12 Middlesbrough 2.52  
## 13 Southampton 3.30  
## 14 Stoke City 1.90  
## 15 Sunderland 3.51  
## 16 Swansea City 3.70  
## 17 Tottenham 2.78  
## 18 Watford 2.33  
## 19 West Bromwich 4.17  
## 20 West Ham Utd 2

summarize(by\_team\_goal, sum(home\_advantage2))

## # A tibble: 20 x 2  
## team `sum(home\_advantage2)`  
## <chr> <dbl>  
## 1 Arsenal 50   
## 2 Bournemouth 50   
## 3 Burnley 100   
## 4 Chelsea 50   
## 5 Crystal Palace 45.5  
## 6 Everton 54.5  
## 7 Hull City 42.9  
## 8 Leicester City 62.5  
## 9 Liverpool 50   
## 10 Manchester City 50   
## 11 Manchester Utd 61.5  
## 12 Middlesbrough 42.9  
## 13 Southampton 42.9  
## 14 Stoke City 40   
## 15 Sunderland 66.7  
## 16 Swansea City 66.7  
## 17 Tottenham 41.7  
## 18 Watford 58.3  
## 19 West Bromwich 62.5  
## 20 West Ham Utd 50

By calculating overall goal\_rate, highest mean goals are scored by Burnley, and lowest mean goals are scored by Stoke City.

Calculate the mean for the number of goals scored by each team group using the aggregate function. There are total 20 teams

aggregate(home\_advantage2 ~ team , Home\_advantage, mean)

## team home\_advantage2  
## 1 Arsenal 50.00000  
## 2 Bournemouth 50.00000  
## 3 Burnley 100.00000  
## 4 Chelsea 50.00000  
## 5 Crystal Palace 45.45455  
## 6 Everton 54.54545  
## 7 Hull City 42.85714  
## 8 Leicester City 62.50000  
## 9 Liverpool 50.00000  
## 10 Manchester City 50.00000  
## 11 Manchester Utd 61.53846  
## 12 Middlesbrough 42.85714  
## 13 Southampton 42.85714  
## 14 Stoke City 40.00000  
## 15 Sunderland 66.66667  
## 16 Swansea City 66.66667  
## 17 Tottenham 41.66667  
## 18 Watford 58.33333  
## 19 West Bromwich 62.50000  
## 20 West Ham Utd 50.00000

By calculating overall mean, highest mean goals are scored by Burnley, and lowest mean goals are scored by Stoke City.

##### Conclusion

Tottenham scored the most points at home, Stoke city scored the least

Burnley scored the most goals at home, Stoke city scored the least.

#### Dataset of Compairing monthly citizenship for a given region

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

#Data from "Tips for Simplifying Crosstab Query Statements"", Rob Gravelle, Database Journal, 2010  
  
citizenship <- read.csv(file="https://raw.githubusercontent.com/yli74/movies/project-2/Crosstab%20Query.csv",header = TRUE,stringsAsFactors = FALSE, check.names = F,sep = ",")  
citizenship

## Month REGION 1 REGION 2 REGION 3 REGION 4 REGION 5 TOTAL  
## 1 April 13 33 76 2 47 171  
## 2 May 17 55 209 1 143 425  
## 3 June 8 63 221 1 127 420  
## 4 July 13 104 240 6 123 486  
## 5 August 18 121 274 9 111 533  
## 6 September 25 160 239 2 88 514  
## 7 October 9 88 295 2 127 521  
## 8 November 2 86 292 2 120 502  
## 9 December 1 128 232 6 155 522  
## 10 TOTAL 106 838 2078 31 1041 4094

##### Tidying and Transforming Data

Using the gather() to transform the data

To tidy the data, there are total 4 varibles in which we can tidy the dataset.

The four variables are: Month, region, month\_total,Total

tidy\_citizenship <- citizenship %>%  
 gather("region","month\_total",2:6) %>%  
 select(Month, region, month\_total,TOTAL)  
tidy\_citizenship

## Month region month\_total TOTAL  
## 1 April REGION 1 13 171  
## 2 May REGION 1 17 425  
## 3 June REGION 1 8 420  
## 4 July REGION 1 13 486  
## 5 August REGION 1 18 533  
## 6 September REGION 1 25 514  
## 7 October REGION 1 9 521  
## 8 November REGION 1 2 502  
## 9 December REGION 1 1 522  
## 10 TOTAL REGION 1 106 4094  
## 11 April REGION 2 33 171  
## 12 May REGION 2 55 425  
## 13 June REGION 2 63 420  
## 14 July REGION 2 104 486  
## 15 August REGION 2 121 533  
## 16 September REGION 2 160 514  
## 17 October REGION 2 88 521  
## 18 November REGION 2 86 502  
## 19 December REGION 2 128 522  
## 20 TOTAL REGION 2 838 4094  
## 21 April REGION 3 76 171  
## 22 May REGION 3 209 425  
## 23 June REGION 3 221 420  
## 24 July REGION 3 240 486  
## 25 August REGION 3 274 533  
## 26 September REGION 3 239 514  
## 27 October REGION 3 295 521  
## 28 November REGION 3 292 502  
## 29 December REGION 3 232 522  
## 30 TOTAL REGION 3 2078 4094  
## 31 April REGION 4 2 171  
## 32 May REGION 4 1 425  
## 33 June REGION 4 1 420  
## 34 July REGION 4 6 486  
## 35 August REGION 4 9 533  
## 36 September REGION 4 2 514  
## 37 October REGION 4 2 521  
## 38 November REGION 4 2 502  
## 39 December REGION 4 6 522  
## 40 TOTAL REGION 4 31 4094  
## 41 April REGION 5 47 171  
## 42 May REGION 5 143 425  
## 43 June REGION 5 127 420  
## 44 July REGION 5 123 486  
## 45 August REGION 5 111 533  
## 46 September REGION 5 88 514  
## 47 October REGION 5 127 521  
## 48 November REGION 5 120 502  
## 49 December REGION 5 155 522  
## 50 TOTAL REGION 5 1041 4094

Summary of the tidy dataset

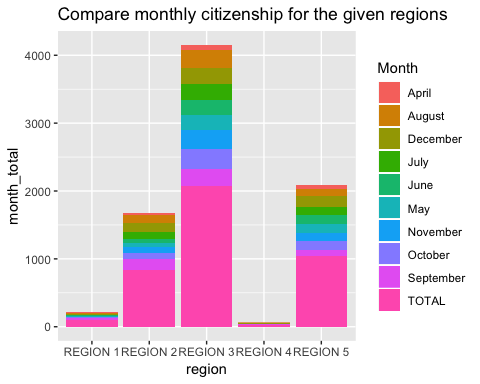
summary(tidy\_citizenship)

## Month region month\_total TOTAL   
## Length:50 Length:50 Min. : 1.0 Min. : 171.0   
## Class :character Class :character 1st Qu.: 10.0 1st Qu.: 425.0   
## Mode :character Mode :character Median : 87.0 Median : 508.0   
## Mean : 163.8 Mean : 818.8   
## 3rd Qu.: 152.0 3rd Qu.: 522.0   
## Max. :2078.0 Max. :4094.0

Now we have a tidy dataset, we’ll Compare monthly citizenship for the given regions by graphics

To see which region issues the most citizenships in the past 9 months

library(ggplot2)  
  
ggplot(data=tidy\_citizenship, aes(x = region, y = month\_total, fill = Month)) + geom\_bar(stat="identity",position = 'stack') + ggtitle('Compare monthly citizenship for the given regions')



Region 3 issued the most citizenships over in the last 9 months and Region 4 issued the least.

Calculate the mean for the number of citizenships issued by each region using the aggregate function. There are total 5 teams.

aggregate(month\_total ~ region ,tidy\_citizenship, mean)

## region month\_total  
## 1 REGION 1 21.2  
## 2 REGION 2 167.6  
## 3 REGION 3 415.6  
## 4 REGION 4 6.2  
## 5 REGION 5 208.2

Therefore, by calculating the overall mean by regions, we conclude that Region 3 issued most citizenships and Region 4 issued least

Now we will summarize the dataset by grouping it by region and calculating the rate in which the citizenship is being offered.

by\_region <- group\_by(tidy\_citizenship, region)   
summarize(by\_region, citizenship\_rate<-sum(month\_total)/sum(TOTAL))

## # A tibble: 5 x 2  
## region `citizenship\_rate <- sum(month\_total)/sum(TOTAL)`  
## <chr> <dbl>  
## 1 REGION 1 0.0259   
## 2 REGION 2 0.205   
## 3 REGION 3 0.508   
## 4 REGION 4 0.00757  
## 5 REGION 5 0.254

summarize(by\_region, sum(month\_total))

## # A tibble: 5 x 2  
## region `sum(month\_total)`  
## <chr> <int>  
## 1 REGION 1 212  
## 2 REGION 2 1676  
## 3 REGION 3 4156  
## 4 REGION 4 62  
## 5 REGION 5 2082

We will carry out a proportion test to know the statistical difference bewttn the Region 3 and Region 4

prop.test(x=c(4156,62), n=c(8188,8188))

##   
## 2-sample test for equality of proportions with continuity  
## correction  
##   
## data: c(4156, 62) out of c(8188, 8188)  
## X-squared = 5349.6, df = 1, p-value < 2.2e-16  
## alternative hypothesis: two.sided  
## 95 percent confidence interval:  
## 0.4888875 0.5111125  
## sample estimates:  
## prop 1 prop 2   
## 0.507572057 0.007572057

We can see there is a significant statistical difference between region 3 and region 4 proportion.

We can also group the dataset by month to make it more specific.

by\_month = group\_by(tidy\_citizenship, Month,region)  
df2 = as.data.frame(summarize(by\_month, citizenship\_rate=sum(month\_total)/sum(TOTAL)))  
df2

## Month region citizenship\_rate  
## 1 April REGION 1 0.076023392  
## 2 April REGION 2 0.192982456  
## 3 April REGION 3 0.444444444  
## 4 April REGION 4 0.011695906  
## 5 April REGION 5 0.274853801  
## 6 August REGION 1 0.033771107  
## 7 August REGION 2 0.227016886  
## 8 August REGION 3 0.514071295  
## 9 August REGION 4 0.016885553  
## 10 August REGION 5 0.208255159  
## 11 December REGION 1 0.001915709  
## 12 December REGION 2 0.245210728  
## 13 December REGION 3 0.444444444  
## 14 December REGION 4 0.011494253  
## 15 December REGION 5 0.296934866  
## 16 July REGION 1 0.026748971  
## 17 July REGION 2 0.213991770  
## 18 July REGION 3 0.493827160  
## 19 July REGION 4 0.012345679  
## 20 July REGION 5 0.253086420  
## 21 June REGION 1 0.019047619  
## 22 June REGION 2 0.150000000  
## 23 June REGION 3 0.526190476  
## 24 June REGION 4 0.002380952  
## 25 June REGION 5 0.302380952  
## 26 May REGION 1 0.040000000  
## 27 May REGION 2 0.129411765  
## 28 May REGION 3 0.491764706  
## 29 May REGION 4 0.002352941  
## 30 May REGION 5 0.336470588  
## 31 November REGION 1 0.003984064  
## 32 November REGION 2 0.171314741  
## 33 November REGION 3 0.581673307  
## 34 November REGION 4 0.003984064  
## 35 November REGION 5 0.239043825  
## 36 October REGION 1 0.017274472  
## 37 October REGION 2 0.168905950  
## 38 October REGION 3 0.566218810  
## 39 October REGION 4 0.003838772  
## 40 October REGION 5 0.243761996  
## 41 September REGION 1 0.048638132  
## 42 September REGION 2 0.311284047  
## 43 September REGION 3 0.464980545  
## 44 September REGION 4 0.003891051  
## 45 September REGION 5 0.171206226  
## 46 TOTAL REGION 1 0.025891549  
## 47 TOTAL REGION 2 0.204689790  
## 48 TOTAL REGION 3 0.507572057  
## 49 TOTAL REGION 4 0.007572057  
## 50 TOTAL REGION 5 0.254274548

##### Conclusion

By doing various analyses on the citizenship dataset, we conclude that:

Region 3 issued the most citizenships over in the last 9 months and Region 4 issued the least.