# PROJECT 2

OMER OZEREN

March 09, 2019

## Table of Contents



---

Choose any of the three wide datasets identified in the week 6 discussion items. (You may choose your own) Read the information from your csv into R and use tidyR and dplyr as needed to transform the data. Perform the analysis requested in the discussion item.

Load Libraries:

```
library(RCurl)
library(stringr)
library(tidyr)
library(dplyr)
library(ggplot2)
```

```
library(psych)
library(knitr)
```

## GINI

Measuring the wealth distribution between the people in each country has been something economists have been measuring for many years. In the GINI index, a higher GINI coefficient signifies inequality in wealth distribution, with 1 being complete inequality and 0 being complete equality.

The World Bank has been maintaining this data.

http://databank.worldbank.org/data/reports.aspx?source=2&series=SI.POV.GINI&country=#

Import dataset from a .csv file.

### Data

```
GINI.rawfile <-
read.csv("https://raw.githubusercontent.com/omerozeren/DATA607/master/PROJECT_2/GINI.csv", header = TRUE)
head(GINI.rawfile)
```

```
##                        Series.Name Series.Code   Country.Name Country.Code
## 1 GINI index (World Bank estimate) SI.POV.GINI    Afghanistan          AFG
## 2 GINI index (World Bank estimate) SI.POV.GINI        Albania          ALB
## 3 GINI index (World Bank estimate) SI.POV.GINI        Algeria          DZA
## 4 GINI index (World Bank estimate) SI.POV.GINI American Samoa          ASM
## 5 GINI index (World Bank estimate) SI.POV.GINI        Andorra          AND
## 6 GINI index (World Bank estimate) SI.POV.GINI         Angola          AGO
##   X1990..YR1990. X2000..YR2000. X2009..YR2009. X2010..YR2010.
## 1             ..             ..             ..             ..
## 2             ..             ..             ..             ..
## 3             ..             ..             ..             ..
## 4             ..             ..             ..             ..
## 5             ..             ..             ..             ..
## 6             ..             52             ..             ..
##   X2011..YR2011. X2012..YR2012. X2013..YR2013. X2014..YR2014.
## 1             ..             ..             ..             ..
## 2             ..             29             ..             ..
## 3           27.6             ..             ..             ..
## 4             ..             ..             ..             ..
## 5             ..             ..             ..             ..
## 6             ..             ..             ..             ..
##   X2015..YR2015. X2016..YR2016. X2017..YR2017. X2018..YR2018.
## 1             ..             ..             ..             ..
## 2             ..             ..             ..             ..
## 3             ..             ..             ..             ..
## 4             ..             ..             ..             ..
```

```
## 5             ..             ..             ..             ..
## 6             ..             ..             ..             ..
```

I notice that from the original data frame, there are columns: 1990, 2000, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016,2017,2018. I will use the gather() function from dplyr to 'tidy' up the data.

### Untidy Data

```
#First,  will rename the columns so that it is easier to read.
colnames(GINI.rawfile) <- c("Series.Name", "Series.Code", "Country.Name",
"Country.Code", 1990, 2000,  2009, 2010, 2011, 2012, 2013, 2014, 2015,
2016,2017,2018)
# Then  will replace ".." with NA
GINI.rawfile[GINI.rawfile == '..'] <- NA
# Next  will gather the data and then eliminate the columns "Series.Name" and
"Series.Code"
GINI <- GINI.rawfile %>% gather(Year, GINI_Index, c(5:16), na.rm = TRUE) %>%
group_by(Country.Name) %>% select(-c(Series.Name, Series.Code)) %>%
arrange(Country.Name)
#   must convert them into numbers so that we can perform calculations and
statistical analysis.
GINI <- transform(GINI, GINI_Index = as.numeric(GINI_Index))
head(GINI)
```

```
##   Country.Name Country.Code Year GINI_Index
## 1                           1990         NA
## 2                           1990         NA
## 3                           1990         NA
## 4                           1990         NA
## 5                           1990         NA
## 6                           2000         NA
```

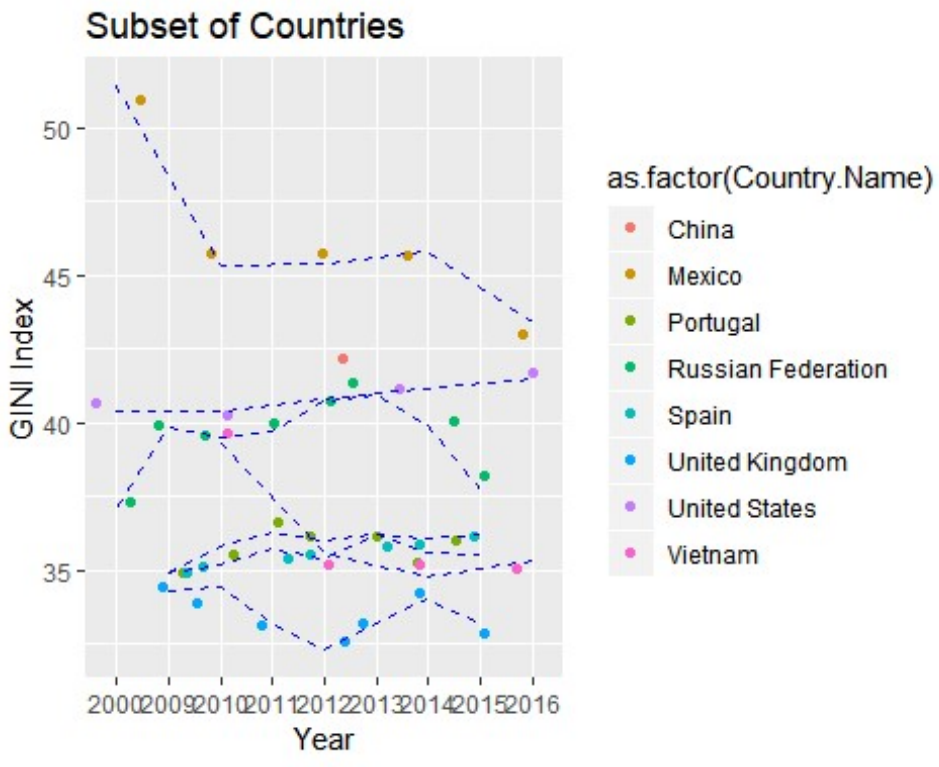
GINI is now formatted into a 'tidy' format that can now be utilized for analysis.

According to the threat, an analysis that could be performed is the trend in the GINI coefficient for each country (or continent) or the average of the GINI coefficient.

Let's demonstrate the trend for the countries GINI coefficient scores.

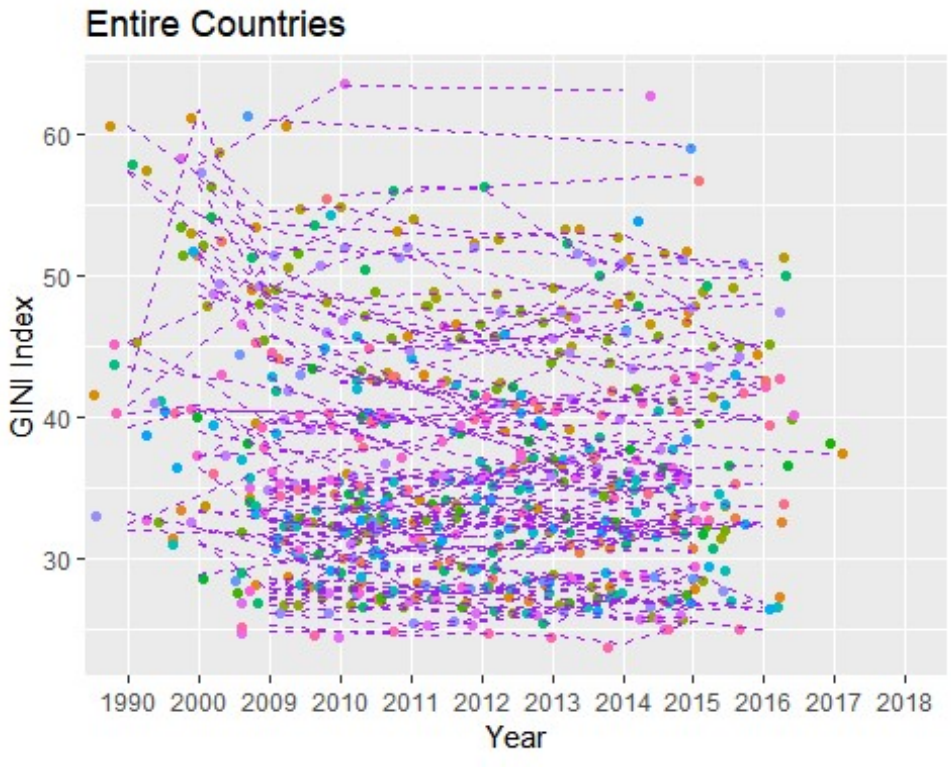
### Subset Data

```
GINI.subset <- GINI %>% filter(Country.Name %in% c('United States',
'Vietnam', 'Spain', 'United Kingdom', 'Turkmenistan', 'Russian Federation',
'Portugal', 'Mexico', 'China'))
ggplot(GINI.subset, aes(x = Year, y = GINI_Index)) + geom_jitter(width = 0.5,
height = 0.5, aes(color = as.factor(Country.Name))) + geom_line(aes(group =
Country.Name), lty = 2, color = "blue") + labs(title = "Subset of Countries",
x = "Year", y = "GINI Index")
```




**Entire countries trend**

```
ggplot(GINI, aes(x = Year, y = GINI_Index)) + geom_jitter(width = 0.5, height = 0.5, aes(color = as.factor(Country.Name))) + geom_line(aes(group = Country.Name), lty = 2, color = "purple") + labs(title = "Entire Countries", x = "Year", y = "GINI Index") + theme(legend.position = "none")
```




### Ranked average GINI Index

```
#  find the max and min of the country's GINI index by dplyr
par(mfrow = c(1,2))
GINI.avg.per.country.order <- GINI %>% group_by(Country.Name) %>%
summarise(AVG_GINI = mean(GINI_Index)) %>% arrange(AVG_GINI)
head(GINI.avg.per.country.order)
```

```
## # A tibble: 6 x 2
##   Country.Name    AVG_GINI
##   <fct>              <dbl>
## 1 Ukraine             24.8
## 2 Slovenia            25.4
## 3 Czech Republic      26.2
## 4 Norway              26.2
## 5 Slovak Republic     26.8
## 6 Iceland             27.0
```

```
GINI.avg.per.country.order.max <- GINI.avg.per.country.order %>%
arrange(desc(AVG_GINI))
head(GINI.avg.per.country.order.max)
```

```
## # A tibble: 6 x 2
##   Country.Name AVG_GINI
##   <fct>           <dbl>
## 1 South Africa     61.4
## 2 Botswana         60.5
## 3 Namibia          60.0
```

```
## 4 Zambia           56.4
## 5 Lesotho          54.2
## 6 Mozambique       54
```

Interestingly, the top 6 counties with the worst GINI indices (higher the number, and hence, worse the inequality) are all in Africa, while the most of the top 6 countries with the best GINI indices are all in Eastern Europe.

```
GINI.avg.per.country <- GINI %>% group_by(Country.Name) %>%
summarise(AVG_GINI = mean(GINI_Index))
GINI.avg.per.country
```

```
## # A tibble: 148 x 2
##    Country.Name AVG_GINI
##    <fct>           <dbl>
##  1 ""               NA
##  2 Albania          29
##  3 Algeria          27.6
##  4 Angola           52
##  5 Argentina        43.3
##  6 Armenia          30.5
##  7 Australia        34.7
##  8 Austria          30.7
##  9 Bangladesh       32.6
## 10 Belarus          27.6
## # ... with 138 more rows
```

### Summary

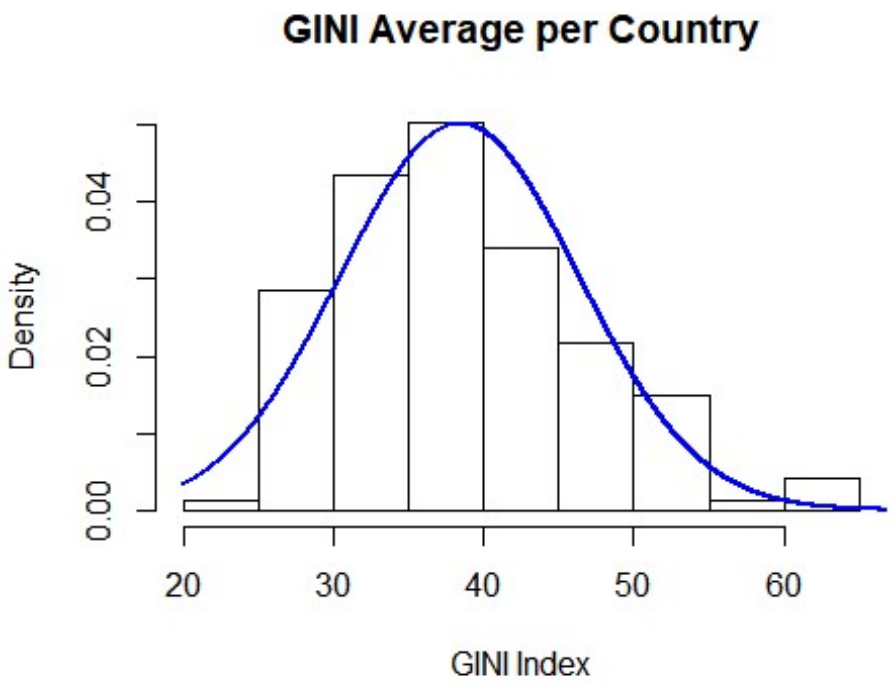
```
GINI.stat <- describe(GINI.avg.per.country$AVG_GINI)
GINI.qq <- summary(GINI.avg.per.country$AVG_GINI)
GINI.stat
```

```
##    vars   n  mean   sd median trimmed  mad   min  max range skew kurtosis
## X1    1 147 38.43 7.95   37.5   37.87 7.56 24.81 61.4 36.59 0.62    -0.06
##      se
## X1 0.66
```

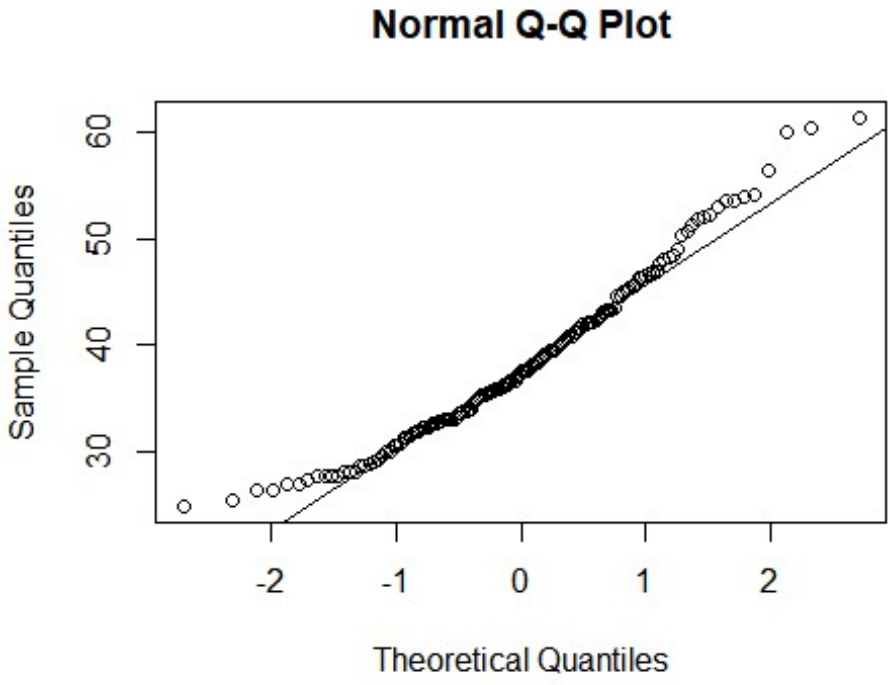
```
GINI.qq
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   24.81   32.72   37.50   38.43   43.10   61.40       1
```

```
hist(GINI.avg.per.country$AVG_GINI, prob = TRUE, main = "GINI Average per
Country", xlab = "GINI Index")
x <- seq(20, 70, length = 10000)
y <- dnorm(x, mean = GINI.stat$mean, sd = GINI.stat$sd)
lines(x, y, type = 'l', lwd = 2, col = 'blue')
```




```r
qqnorm(GINI.avg.per.country$AVG_GINI)
qqline(GINI.avg.per.country$AVG_GINI)
```



```
# Let's find the GINI mean index value for United States
United_States.GINI <- GINI.avg.per.country[GINI.avg.per.country$Country.Name
== 'United States', 'AVG_GINI']
paste0("United_States Average GINI index: ", round(United_States.GINI, 2))

## [1] "United_States Average GINI index: 40.83"

# Calculate the z-score and percentile.
world.mean <- mean(GINI.avg.per.country$AVG_GINI,na.rm = T)
United_States.Z <- (United_States.GINI - world.mean)/GINI.stat$sd
United_States.prob <- pnorm(United_States.Z$AVG_GINI, mean = 0, sd = 1)
paste("United States is", round(United_States.Z$AVG_GINI,2), "standard
deviations away from the mean and is", round(United_States.prob, 2) * 100,
"percentile.")

## [1] "United States is 0.3 standard deviations away from the mean and is 62
percentile."
```

SUMMARY : The United Sates has balance out the wealth and equality . The United States is far only 0.3 standard deviation from world mean "GINI SCORE" and it is on 62 percentile.

## POPULATION

This data shows changes in population by counties from 1960 to 2017. Here I'm going to define that there have been population shifts and I want to highlight some of them in my analysis in a very easy to read visualization.

The World Bank has been maintaining this data.

https://data.worldbank.org/indicator/sp.pop.totl

Import dataset from a .csv file.

### Data

```
population<-
read.csv("https://raw.githubusercontent.com/omerozeren/DATA607/master/PROJECT
_2/populationbycountry.csv", header = TRUE)
population[1:15, 1:5]

##            Country.Name    X1960    X1961    X1962     X1963
## 1                 Aruba    54211    55438    56225     56695
## 2           Afghanistan  8996351  9166764  9345868   9533954
## 3                Angola  5643182  5753024  5866061   5980417
## 4               Albania  1608800  1659800  1711319   1762621
## 5               Andorra    13411    14375    15370     16412
## 6            Arab World 92490932 95044497 97682294 100411076
## 7  United Arab Emirates    92634   101078   112472    125566
## 8             Argentina 20619075 20953077 21287682  21621840
## 9               Armenia  1874120  1941491  2009526   2077575
## 10       American Samoa    20013    20486    21117     21882
## 11  Antigua and Barbuda    55339    56144    57144     58294
```

```
## 12            Australia 10276477 10483000 10742000  10950000
## 13              Austria  7047539  7086299  7129864   7175811
## 14           Azerbaijan  3895396  4030320  4171425   4315128
## 15              Burundi  2786106  2839666  2893669   2949926
```

I want to remove Country.Names that actually not country such as

-North America -Central & South America -Antarctica -Eurasia -Middle East -Asia & Oceania -World -Africa -Europe -Former Czechoslovakia -Former Serbia and Montenegro -Former Yugoslavia -East -Hawaiian Trade Zone -U.S. Pacific Islands -Wake Island -Former U.S.S.R.

```
kable(head(population$Country.Name,10))
```

| x |
|---|
| Aruba |
| Afghanistan |
| Angola |
| Albania |
| Andorra |
| Arab World |
| United Arab Emirates |
| Argentina |
| Armenia |
| American Samoa |

Make a vector that lists all the countries that could not be classified by country

```
remove<- c('North America',
           'Central & South America',
           'Antarctica',
           'Eurasia',
           'Middle East',
           'Asia & Oceania',
           'World', 'Africa', 'Europe',
           'Former Czechoslovakia',
           'Former Serbia and Montenegro',
           'Former Yugoslavia',
           'East', 'Hawaiian Trade Zone',
           'U.S. Pacific Islands', 'Wake Island', 'Former U.S.S.R.',
           'IDA & IBRD total','Low & middle income',
'Middle income','IBRD only','Early-demographic dividend','Upper middle
income','Lower middle income',',Late-demographic dividend',
'South Asia','South Asia (IDA & IBRD)','OECD members','High income','Post-
demographic dividend','IDA total','IDA only',
'Least developed countries: UN classification','Pre-demographic
dividend','Latin America & Caribbean','Latin America & the Caribbean (IDA &
IBRD countries)','Heavily indebted poor countries (HIPC)',
```

```
'Low income','Latin America & Caribbean (excluding high income)','Euro
area','IDA blend','Fragile and conflict affected situations','Late-
demographic dividend',
'Latin America & the Caribbean (IDA & IBRD countries)','Heavily indebted poor
countries (HIPC)','Latin America & the Caribbean (IDA & IBRD
countries)','Heavily indebted poor countries (HIPC)')
df <- population[ !grepl(paste(remove, collapse="|"),
population$Country.Name),]
df <- data.frame(df)
head(df)
```

```
##           Country.Name   X1960   X1961   X1962   X1963   X1964   X1965
## 1                Aruba   54211   55438   56225   56695   57032   57360
## 2          Afghanistan 8996351 9166764 9345868 9533954 9731361 9938414
## 3               Angola 5643182 5753024 5866061 5980417 6093321 6203299
## 4              Albania 1608800 1659800 1711319 1762621 1814135 1864791
## 5              Andorra   13411   14375   15370   16412   17469   18549
## 7 United Arab Emirates   92634  101078  112472  125566  138529  150362
##      X1966    X1967    X1968    X1969    X1970    X1971    X1972    X1973
## 1    57715    58055    58386    58726    59063    59440    59840    60243
## 2 10152331 10372630 10604346 10854428 11126123 11417825 11721940 12027822
## 3  6309770  6414995  6523791  6642632  6776381  6927269  7094834  7277960
## 4  1914573  1965598  2022272  2081695  2135479  2187853  2243126  2296752
## 5    19647    20758    21890    23058    24276    25559    26892    28232
## 7   160481   170283   183194   203820   235499   278808   332760   397174
##      X1974    X1975    X1976    X1977    X1978    X1979    X1980    X1981
## 1    60528    60657    60586    60366    60103    59980    60096    60567
## 2 12321541 12590286 12840299 13067538 13237734 13306695 13248370 13053954
## 3  7474338  7682479  7900997  8130988  8376147  8641521  8929900  9244507
## 4  2350124  2404831  2458526  2513546  2566266  2617832  2671997  2726056
## 5    29520    30705    31777    32771    33737    34818    36067    37500
## 7   471364   554324   646943   748117   852262   952040  1042384  1120900
##      X1982    X1983    X1984    X1985    X1986    X1987    X1988    X1989
## 1    61345    62201    62836    63026    62644    61833    61079    61032
## 2 12749645 12389269 12047115 11783050 11601041 11502761 11540888 11777609
## 3  9582156  9931562 10277321 10609042 10921037 11218268 11513968 11827237
## 4  2784278  2843960  2904429  2964762  3022635  3083605  3142336  3227943
## 5    39114    40867    42706    44600    46517    48455    50434    52448
## 7  1189545  1253060  1318478  1391052  1472218  1560718  1655849  1756043
##      X1990    X1991    X1992    X1993    X1994    X1995    X1996    X1997
## 1    62149    64622    68235    72504    76700    80324    83200    85451
## 2 12249114 12993657 13981231 15095099 16172719 17099541 17822884 18381605
## 3 12171441 12553446 12968345 13403734 13841301 14268994 14682284 15088981
## 4  3286542  3266790  3247039  3227287  3207536  3187784  3168033  3148281
## 5    54509    56671    58888    60971    62677    63850    64360    64327
## 7  1860174  1970026  2086639  2207405  2328686  2448820  2571020  2700010
##      X1998    X1999    X2000    X2001    X2002    X2003    X2004    X2005
## 1    87277    89005    90853    92898    94992    97017    98737   100031
## 2 18863999 19403676 20093756 20966463 21979923 23064851 24118979 25070798
## 3 15504318 15949766 16440924 16983266 17572649 18203369 18865716 19552542
```

```
## 4  3128530  3108778  3089027  3060173  3051010  3039616  3026939  3011487
## 5    64142    64370    65390    67341    70049    73182    76244    78867
## 7  2838145  2988162  3154925  3326032  3507232  3741932  4087931  4579562
##      X2006    X2007    X2008    X2009    X2010    X2011    X2012    X2013
## 1   100832   101220   101353   101453   101669   102053   102577   103187
## 2 25893450 26616792 27294031 28004331 28803167 29708599 30696958 31731688
## 3 20262399 20997687 21759420 22549547 23369131 24218565 25096150 25998340
## 4  2992547  2970017  2947314  2927519  2913021  2905195  2900401  2895092
## 5    80991    82683    83861    84462    84449    83751    82431    80788
## 7  5242032  6044067  6894278  7666393  8270684  8672475  8900453  9006263
##      X2014    X2015    X2016    X2017
## 1   103795   104341   104822   105264
## 2 32758020 33736494 34656032 35530081
## 3 26920466 27859305 28813463 29784193
## 4  2889104  2880703  2876101  2873457
## 5    79223    78014    77281    76965
## 7  9070867  9154302  9269612  9400145
```

I need to clean all of my year columns have an X in front of the name. We can use some regular expression to clean the column names.

```
#remove the x
names(df) <- gsub(x = names(df), pattern = "\\X", replacement = "")
names(df)
```

```
##  [1] "Country.Name" "1960"         "1961"         "1962"
##  [5] "1963"         "1964"         "1965"         "1966"
##  [9] "1967"         "1968"         "1969"         "1970"
## [13] "1971"         "1972"         "1973"         "1974"
## [17] "1975"         "1976"         "1977"         "1978"
## [21] "1979"         "1980"         "1981"         "1982"
## [25] "1983"         "1984"         "1985"         "1986"
## [29] "1987"         "1988"         "1989"         "1990"
## [33] "1991"         "1992"         "1993"         "1994"
## [37] "1995"         "1996"         "1997"         "1998"
## [41] "1999"         "2000"         "2001"         "2002"
## [45] "2003"         "2004"         "2005"         "2006"
## [49] "2007"         "2008"         "2009"         "2010"
## [53] "2011"         "2012"         "2013"         "2014"
## [57] "2015"         "2016"         "2017"
```

```
head(df, 2)
```

```
##   Country.Name    1960    1961    1962    1963    1964    1965     1966
## 1        Aruba   54211   55438   56225   56695   57032   57360    57715
## 2  Afghanistan 8996351 9166764 9345868 9533954 9731361 9938414 10152331
##       1967     1968     1969     1970     1971     1972     1973     1974
## 1    58055    58386    58726    59063    59440    59840    60243    60528
## 2 10372630 10604346 10854428 11126123 11417825 11721940 12027822 12321541
##       1975     1976     1977     1978     1979     1980     1981     1982
## 1    60657    60586    60366    60103    59980    60096    60567    61345
```

```
## 2 12590286 12840299 13067538 13237734 13306695 13248370 13053954 12749645
##       1983     1984     1985     1986     1987     1988     1989     1990
## 1    62201    62836    63026    62644    61833    61079    61032    62149
## 2 12389269 12047115 11783050 11601041 11502761 11540888 11777609 12249114
##       1991     1992     1993     1994     1995     1996     1997     1998
## 1    64622    68235    72504    76700    80324    83200    85451    87277
## 2 12993657 13981231 15095099 16172719 17099541 17822884 18381605 18863999
##       1999     2000     2001     2002     2003     2004     2005     2006
## 1    89005    90853    92898    94992    97017    98737   100031   100832
## 2 19403676 20093756 20966463 21979923 23064851 24118979 25070798 25893450
##       2007     2008     2009     2010     2011     2012     2013     2014
## 1   101220   101353   101453   101669   102053   102577   103187   103795
## 2 26616792 27294031 28004331 28803167 29708599 30696958 31731688 32758020
##       2015     2016     2017
## 1   104341   104822   105264
## 2 33736494 34656032 35530081

#Lets gather by key value pairs and create a new data frame
total_pop <- df %>% gather(data=df, Population, "1960":"2017")
head(total_pop, 10)

##            Country.Name    . Population
## 1                 Aruba 1960      54211
## 2           Afghanistan 1960    8996351
## 3                Angola 1960    5643182
## 4               Albania 1960    1608800
## 5               Andorra 1960      13411
## 6  United Arab Emirates 1960      92634
## 7             Argentina 1960   20619075
## 8               Armenia 1960    1874120
## 9        American Samoa 1960      20013
## 10  Antigua and Barbuda 1960      55339

tail(total_pop, 10)

##                 Country.Name    . Population
## 12867          Venezuela, RB 2017   31977065
## 12868 British Virgin Islands 2017      31196
## 12869  Virgin Islands (U.S.) 2017     107268
## 12870                Vietnam 2017   95540800
## 12871                Vanuatu 2017     276244
## 12872                  Samoa 2017     196440
## 12873                 Kosovo 2017    1830700
## 12874            Yemen, Rep. 2017   28250420
## 12875                 Zambia 2017   17094130
## 12876               Zimbabwe 2017   16529904
```

I need to rename the year column

```
colnames(total_pop)[colnames(total_pop)=="."]<-"Year"
names(total_pop)
```

```
## [1] "Country.Name" "Year"         "Population"
```

removing the Na rows

```
total_pop<-na.omit(total_pop)
head(total_pop)
```

```
##           Country.Name Year Population
## 1                Aruba 1960      54211
## 2          Afghanistan 1960    8996351
## 3               Angola 1960    5643182
## 4              Albania 1960    1608800
## 5              Andorra 1960      13411
## 6 United Arab Emirates 1960      92634
```

### Subset Population

In subset population data, I choose Turkey to see how population is look like.

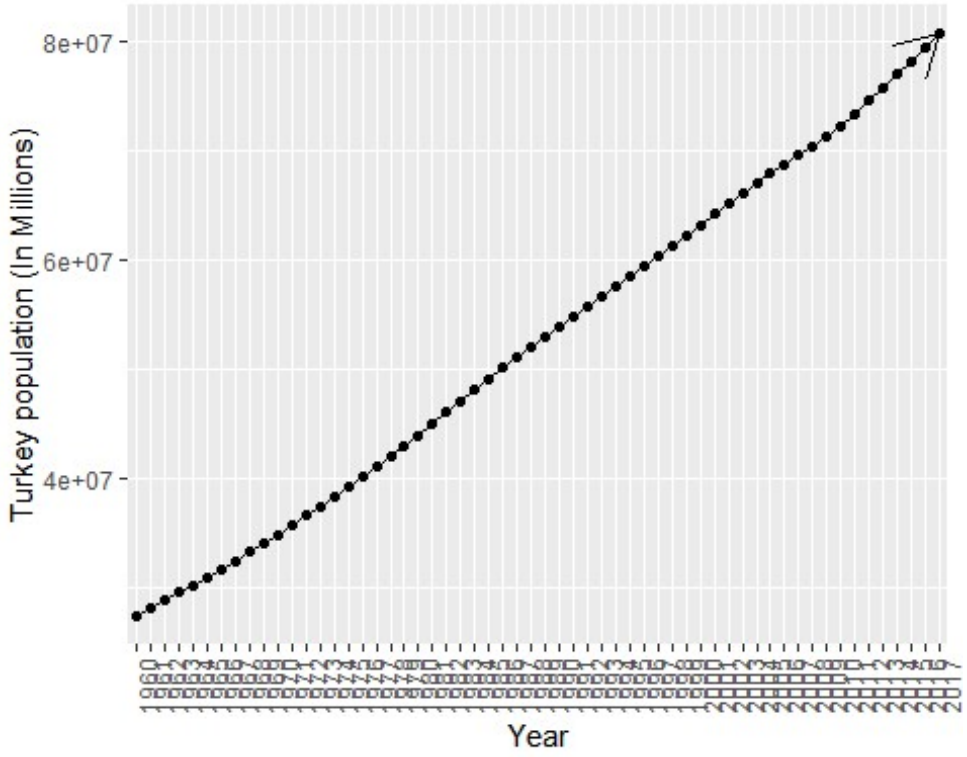
Creating Subset

```
df.Turkey<-subset(total_pop, Country.Name=='Turkey', select=c(Country.Name,
Year, Population))
head(df.Turkey, 30)
```

```
##      Country.Name Year Population
## 204        Turkey 1960   27472331
## 426        Turkey 1961   28146893
## 648        Turkey 1962   28832805
## 870        Turkey 1963   29531342
## 1092       Turkey 1964   30244232
## 1314       Turkey 1965   30972965
## 1536       Turkey 1966   31717477
## 1758       Turkey 1967   32477961
## 1980       Turkey 1968   33256432
## 2202       Turkey 1969   34055361
## 2424       Turkey 1970   34876267
## 2646       Turkey 1971   35720568
## 2868       Turkey 1972   36587225
## 3090       Turkey 1973   37472298
## 3312       Turkey 1974   38370241
## 3534       Turkey 1975   39277211
## 3756       Turkey 1976   40189511
## 3978       Turkey 1977   41108248
## 4200       Turkey 1978   42039935
## 4422       Turkey 1979   42993991
## 4644       Turkey 1980   43975921
## 4866       Turkey 1981   44988356
## 5088       Turkey 1982   46025357
## 5310       Turkey 1983   47073422
## 5532       Turkey 1984   48114105
```

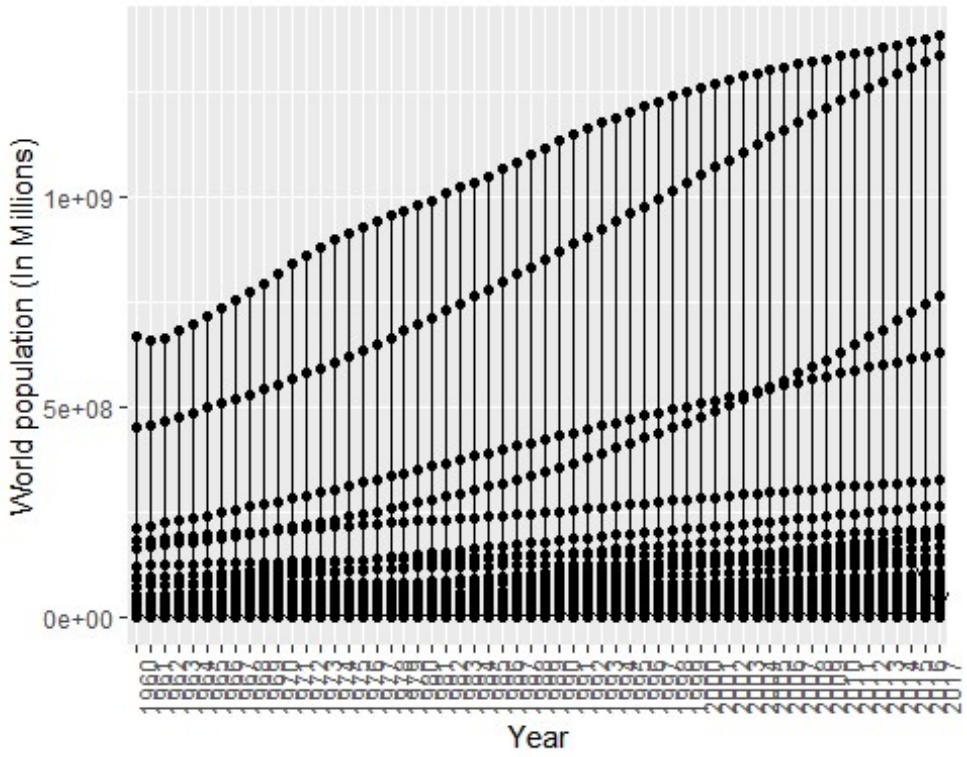
```
## 5754       Turkey 1985   49133883
## 5976       Turkey 1986   50128489
## 6198       Turkey 1987   51100878
## 6420       Turkey 1988   52053704
## 6642       Turkey 1989   52992429

#Visualize the population
ggplot(data=df.Turkey, aes(x=Year, y=Population, group=1)) +
  geom_line(arrow = arrow())+
  geom_point()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  labs( x="Year", y="Turkey population (In Millions)")
```




## Entire country Population

```
#Visualize the population
ggplot(data=total_pop, aes(x=Year, y=Population, group=1)) +
  geom_line(arrow = arrow())+
  geom_point()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  labs( x="Year", y="World population (In Millions)")
```




## Untidy Data

```
#  find the max and min of the country's Population index by dplyr
par(mfrow = c(1,2))
POP.avg.per.country.order <- total_pop %>% group_by(Year)  %>%
group_by(Country.Name) %>% summarise(AVG_POP =
mean(Population,na.rm=TRUE))%>% arrange(AVG_POP)
head(POP.avg.per.country.order)
```

```
## # A tibble: 6 x 2
##   Country.Name             AVG_POP
##   <fct>                      <dbl>
## 1 Nauru                      8568.
## 2 Tuvalu                     8745
## 3 Turks and Caicos Islands  15170.
## 4 Palau                     15479.
## 5 British Virgin Islands    16759.
## 6 St. Martin (French part)  19246.
```

```
POP.avg.per.country.order.max <- POP.avg.per.country.order %>%
arrange(desc(AVG_POP))
head(POP.avg.per.country.order.max)
```

```
## # A tibble: 6 x 2
##   Country.Name                                          AVG_POP
##   <fct>                                                   <dbl>
## 1 China                                                1077271293.
## 2 India                                                 861643130.
```

```
## 3 Latin America & the Caribbean (IDA & IBRD countries)  417789480.
## 4 Heavily indebted poor countries (HIPC)                384511872.
## 5 United States                                         251274672.
## 6 Indonesia                                             174988161.
```

### Summary

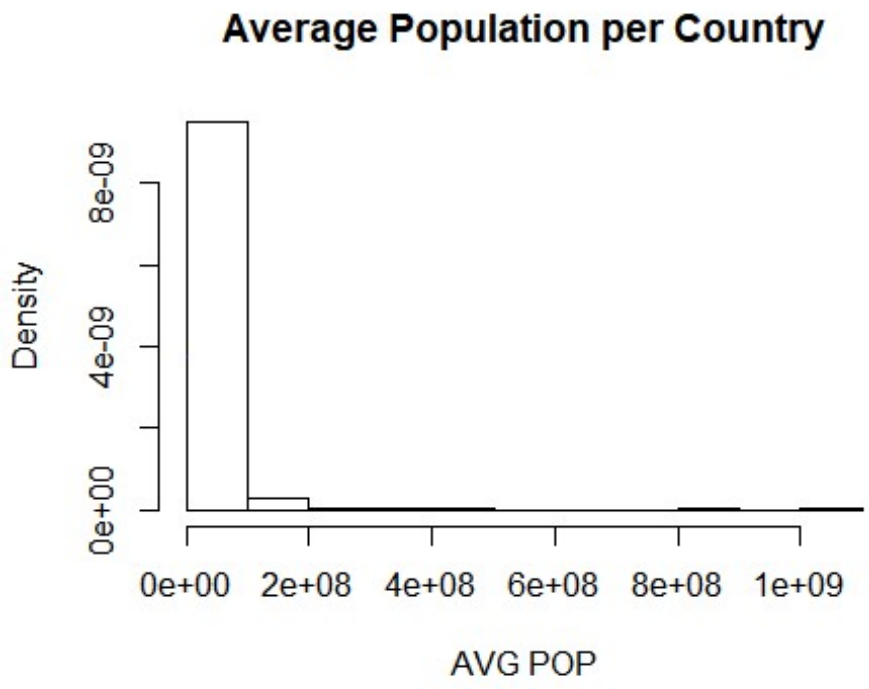
```
POP.stat <- describe(POP.avg.per.country.order.max$AVG_POP)
POP.qq <- summary(POP.avg.per.country.order.max$AVG_POP)
(POP.stat)

##    vars   n     mean        sd  median trimmed     mad    min        max
## X1    1 221 27063439 102588210 4269193 8335631 6179046 8567.5 1077271293
##         range skew kurtosis      se
## X1 1077262726 7.81    68.28 6900830

POP.qq

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 8.568e+03 5.217e+05 4.269e+06 2.706e+07 1.473e+07 1.077e+09

hist(POP.avg.per.country.order.max$AVG_POP, prob = TRUE, main = "Average
Population per Country", xlab = "AVG POP")
y <- dnorm(x, mean = POP.stat$mean, sd = POP.stat$sd)
lines(x, y, type = 'l', lwd = 2, col = 'blue')
```



The summary table shows that the average min population is 8567.5 and average maximum poulation is 422187266. The country that has maximun average population is China and second is India so on.

## LOTTERY

The New York State Government keeps track of all the drawn winning numbers. This database contains information from June 2014 to March 2019.

Defination : Each game costs $2. Players choose (or have the terminal select the numbers, which is known as “quick pick” in Maryland, New Jersey, New York, Pennsylvania, and Tennessee; and “easy pick” in Virginia) 5 of 60 numbers in the main field, and 1 of 4 (hence the game’s name) green “Cash Ball” numbers in a second field. Matching all six numbers wins, or shares (“split-prize liability”).

Source Defination : https://en.wikipedia.org/wiki/New_York_Lottery

Data Source : https://data.ny.gov/Government-Finance/Lottery-Cash-4-Life-Winning-Numbers-Beginning-2014/kwxv-fwze

### Data

```
# Reading the .csv file from my github page
lottery <-
read.csv("https://raw.githubusercontent.com/omerozeren/DATA607/master/PROJECT_2/LOTTERY.csv", header = TRUE)
# This shows the first 6 rows of this data.frame
head(lottery)
```

```
##   Draw.Date Winning.Numbers Cash.Ball
## 1  3/7/2019  07 14 20 38 58         1
## 2  3/4/2019  06 09 45 49 55         4
## 3 2/28/2019  03 15 18 21 35         2
## 4 2/25/2019  18 24 42 55 58         3
## 5 2/21/2019  03 04 34 38 39         2
## 6 2/18/2019  01 37 39 48 54         4
```

### Untidy Data

```
# We will separate all the winning numbers and create separate columns i.e. in row 1, Ball 1: 09, Ball 2: 36, Ball 3: 44, Ball 4: 53, Ball 5: 59
lottery.separated <- unlist(str_extract_all(lottery$Winning.Numbers, "(\\d)."))
lottery.separated <- as.numeric(lottery.separated)
lottery2 <- matrix(lottery.separated, ncol = 5, byrow = TRUE)
lottery2 <- as.data.frame(lottery2)
lottery.untidy <- data.frame(lottery$Draw.Date, lottery2, lottery$Cash.Ball)
colnames(lottery.untidy) <- c("Draw.Date", 1, 2, 3, 4, 5, "Cash.Ball")
# As you can see, this separated all the numbers into its own fields
head(lottery.untidy)
```

```
##   Draw.Date  1  2  3  4  5 Cash.Ball
## 1  3/7/2019  7 14 20 38 58         1
## 2  3/4/2019  6  9 45 49 55         4
## 3 2/28/2019  3 15 18 21 35         2
## 4 2/25/2019 18 24 42 55 58         3
## 5 2/21/2019  3  4 34 38 39         2
## 6 2/18/2019  1 37 39 48 54         4
```

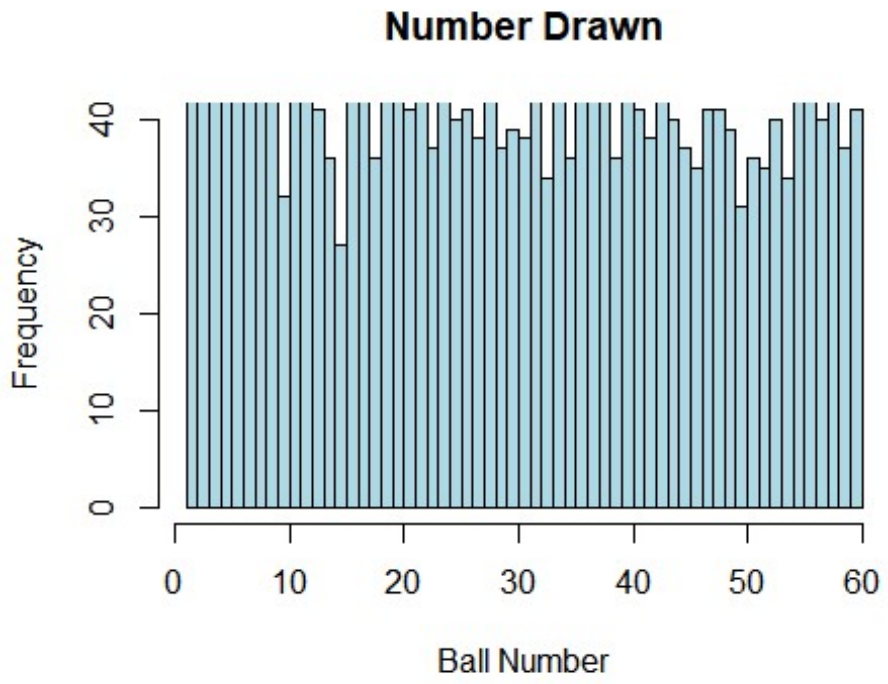
So now we have created an untidy data frame with all of the Cash 4 Life Winning Numbers. We will now utilize dplyr and tidyr to 'tidy' up the data.

```
lottery.tidy <- lottery.untidy %>% gather(BallOrder, Number, 2:6) %>%
select(-Cash.Ball)
head(lottery.tidy)
```

```
##   Draw.Date BallOrder Number
## 1  3/7/2019         1      7
## 2  3/4/2019         1      6
## 3 2/28/2019         1      3
## 4 2/25/2019         1     18
## 5 2/21/2019         1      3
## 6 2/18/2019         1      1
```

Next, I would liket to see 1. What is the most frequent ball to show up? 2. What is the least frequent ball to show up?

```
hist(lottery.tidy$Number, breaks = 60, main = "Number Drawn", xlab = "Ball
Number", ylim = c(0,40), col = 'lightblue')
```




```
Ball.Num <- as.numeric(lottery.tidy$Number)
Ball.Num <- as.data.frame(table(Ball.Num))
head(Ball.Num)
```

```
##   Ball.Num Freq
## 1        1   41
## 2        2   36
## 3        3   42
## 4        4   52
## 5        5   47
## 6        6   43
```

Now with the frequency calculated for each number of ball. Let's find out some more information about these ball numbers.

```
Freq.Drawn <- Ball.Num %>% arrange(desc(Freq))
Freq.Drawn
```

```
##    Ball.Num Freq
## 1         4   52
## 2        55   52
## 3         8   49
## 4        24   49
## 5        38   49
## 6        11   48
## 7        28   48
## 8        37   48
```

```
## 9        40   48
## 10       43   48
## 11        5   47
## 12       20   47
## 13        9   46
## 14       12   45
## 15       56   45
## 16        7   44
## 17       17   44
## 18       58   44
## 19        6   43
## 20       19   43
## 21       22   43
## 22       32   43
## 23        3   42
## 24       16   42
## 25       34   42
## 26       36   42
## 27        1   41
## 28       13   41
## 29       21   41
## 30       26   41
## 31       41   41
## 32       47   41
## 33       48   41
## 34       60   41
## 35       25   40
## 36       44   40
## 37       53   40
## 38       57   40
## 39       30   39
## 40       49   39
## 41       27   38
## 42       31   38
## 43       42   38
## 44       23   37
## 45       29   37
## 46       45   37
## 47       59   37
## 48        2   36
## 49       14   36
## 50       18   36
## 51       35   36
## 52       39   36
## 53       51   36
## 54       46   35
## 55       52   35
## 56       33   34
## 57       54   34
## 58       10   32
```

```
## 59       50   31
## 60       15   27
```

```
paste("The Most Drawn Ball is: ", Freq.Drawn$Ball.Num[1])
```

```
## [1] "The Most Drawn Ball is:  4"
```

```
paste("The Least Drawn Ball is: ",
Freq.Drawn$Ball.Num[length(Freq.Drawn$Ball.Num)])
```

```
## [1] "The Least Drawn Ball is:  15"
```

```
Ball.stat <- Freq.Drawn %>% summarise(Mean= mean(Freq))
Ball.stat <- as.numeric(Ball.stat)
paste("A number was drawn on average: ", Ball.stat)
```

```
## [1] "A number was drawn on average:  41.0833333333333"
```

### Summary

So at least from the histogram, the majority of the numbers appear to be drawn fairly evenly. But as you can see, ball 4 comes very often and ball 15 does not over the course of the 3 years.

I'll make a null and alternative hypothesis. Utilizing the 2 tailed p value test (using alpha = 0.05), we will determine if there is a statistical anomaly.

The null hypothesis is that the number 4 and 15 are variance and are not statistical outliers.

The alternative hypothesis is that the NY State Lottery system is rigged and weighs number 4 and 15 differently from the rest of the numbers.

```
# Calculating the standard deviation, which we will use to calculate the Z-
score
Ball.sd <- sd(Freq.Drawn$Freq)
# The Z-score is obtained and coverted into a two-taled p value Z.4.p
Z.4 <- (60 - Ball.stat)/Ball.sd
Z.4 <- pnorm(Z.4, mean = 0, sd = 1)
Z.4.p <- (1 - Z.4) * 2
# Again, the same process takes place here, except that this time, it is for
ball 42
Z.15 <- (9 - Ball.stat)/Ball.sd
Z.15 <- pnorm(Z.15, mean = 0, sd = 1)
Z.15.p <- (Z.15) * 2
paste("The P-value for Ball 4: ", Z.4.p)
```

```
## [1] "The P-value for Ball 4:  0.000354100249031708"
```

```
paste("The P-value for Ball 15: ", Z.15.p)
```

```
## [1] "The P-value for Ball 15:  1.37443761802246e-09"
```

Both these values are less than alpha = 0.05, thus making them statistically significant. Now does this actually mean that the NYS Lottery System is rigged? The numbers make a strong case for rejecting the null hypothesis.

Even though the p-values suggest that we reject the null hypothesis, we have to remember that this lottery has only been played 494 times since June 2014.That truelly means that our analysis is based on only 494 data points so conclusion might not be accurate.