

STATS 205: Final Project Write-Up

Brian Liu

6/14/2019

1. Background of the data and why it is interesting or important

The data we are using is the data from WHO suicide statistics from Kaggle. This gives population-based statistics on suicide rate.

The reason this data is interesting and important is that suicide is prevalent in many times and places around the world, but many places and times have different suicide rates. When it comes to suicide, there are many potential factors or attributes that may be correlated with an increased risk of suicide, such as:

- a person's sex
- the age group a person belongs to
- the generation a person was born in

The goal is to find significant correlations between these factors and suicide rates: that is, does x factor positively predict suicide rate?

The simple inspiration is suicide prevention: If we can identify the factors that correlate positively with, or predict high suicide rates, then we can target our suicide prevention efforts towards populations with those high-risk factors or attributes.

2. Explanation of the method studied and its properties

We will use the statistical techniques of nonparametric bootstrap and parametric bootstrap methods to aid in prediction, with linear regression as well, and use cross-validation to test if, given new data for a population, this population is at risk of suicide. In other words, predict if the suicide rate would be abnormally or significantly high, and then compare the performance between the two methods (nonparametric and parametric).

Bootstrapping

In statistics, bootstrapping is any test or metric that relies on random sampling with replacement. Bootstrapping allows assigning measures of accuracy (defined in terms of bias, variance, confidence intervals, prediction error or some other such measure) to sample estimates (Efron and Tibshirani 1993; Efron 2003). This technique allows estimation of the sampling distribution of almost any statistic using random sampling methods. Generally, it falls in the broader class of resampling methods.

Bootstrapping is the practice of estimating properties of an estimator (such as its variance) by measuring those properties when sampling from an approximating distribution. One standard choice for an approximating distribution is the empirical distribution function of the observed data. In the case where a set of observations can be assumed to be from an independent and identically distributed population, this can be implemented by constructing a number of resamples with replacement, of the observed dataset (and of equal size to the observed dataset).

It may also be used for constructing hypothesis tests. It is often used as an alternative to statistical inference based on the assumption of a parametric model when that assumption is in doubt, or

where parametric inference is impossible or requires complicated formulas for the calculation of standard errors.

Nonparametric vs. Parametric bootstrap

Linear regression - Kendall rank correlation coefficient

In statistics, the Kendall rank correlation coefficient, commonly referred to as Kendall's tau coefficient (after the Greek letter τ), is a statistic used to measure the ordinal association between two measured quantities. A tau test is a non-parametric hypothesis test for statistical dependence based on the tau coefficient.

It is a measure of rank correlation: the similarity of the orderings of the data when ranked by each of the quantities. It is named after Maurice Kendall, who developed it in 1938,[1] though Gustav Fechner had proposed a similar measure in the context of time series in 1897.[2]

Intuitively, the Kendall correlation between two variables will be high when observations have a similar (or identical for a correlation of 1) rank (i.e. relative position label of the observations within the variable: 1st, 2nd, 3rd, etc.) between the two variables, and low when observations have a dissimilar (or fully different for a correlation of -1) rank between the two variables.

Both Kendall's τ and Spearman's ρ can be formulated as special cases of a more general correlation coefficient.

Cross validation

3. Data analysis or simulation study

We will use the crude rate of suicide per 100,000 people.

This analysis provides information on age-standardized rates...

```
who_suicide_statistics_df <- read.csv("who_suicide_statistics.csv")
head(who_suicide_statistics_df)
```

```
##   country year    sex      age suicides_no population
## 1 Albania 1985 female 15-24 years         NA      277900
## 2 Albania 1985 female 25-34 years         NA      246800
## 3 Albania 1985 female 35-54 years         NA      267500
## 4 Albania 1985 female  5-14 years         NA      298300
## 5 Albania 1985 female 55-74 years         NA      138700
## 6 Albania 1985 female  75+ years         NA       34200
```

```
colnames(who_suicide_statistics_df)
```

```
## [1] "country"    "year"       "sex"        "age"        "suicides_no"
## [6] "population"
```

Filter and save countries with missing suicide rate.

```
library(tidyverse)
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method      from
## [.quosures    rlang
## c.quosures     rlang
```

```
## print.quosures rlang
## -- Attaching packages ----- tidyverse 1.2.1 --
## v ggplot2 3.1.1      v purrr 0.3.2
## v tibble 2.1.1       v dplyr 0.8.1
## 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()

filtered_suicide_df <- drop_na(who_suicide_statistics_df, "suicides_no")
head(filtered_suicide_df)
```

```
##   country year  sex      age suicides_no population
## 25 Albania 1987 female 15-24 years         14      289700
## 26 Albania 1987 female 25-34 years          4      257200
## 27 Albania 1987 female 35-54 years          6      278800
## 28 Albania 1987 female 5-14 years           0      311000
## 29 Albania 1987 female 55-74 years          0      144600
## 30 Albania 1987 female 75+ years            1       35600
```

After filtering countries with missing suicide rate, take a random sample of 100 countries and make sure each continent has approximately equal countries.

Filter countries by continent:

```
library(countrycode)
filtered_suicide_df$continent <- countrycode(sourcevar = filtered_suicide_df[, "country"],
      origin = "country.name",
      destination = "continent")
```

```
## Warning in countrycode(sourcevar = filtered_suicide_df[, "country"], origin = "country.name", : Some
## Warning in countrycode(sourcevar = filtered_suicide_df[, "country"], origin = "country.name", : Some
head(filtered_suicide_df)
```

```
##   country year  sex      age suicides_no population continent
## 25 Albania 1987 female 15-24 years         14      289700   Europe
## 26 Albania 1987 female 25-34 years          4      257200   Europe
## 27 Albania 1987 female 35-54 years          6      278800   Europe
## 28 Albania 1987 female 5-14 years           0      311000   Europe
## 29 Albania 1987 female 55-74 years          0      144600   Europe
## 30 Albania 1987 female 75+ years            1       35600   Europe

write.csv(filtered_suicide_df, 'filtered_suicide.csv')
```

Let us find out which continents are counted:

```
# Get list of continents
list_of_continents <- unique(filtered_suicide_df$continent); list_of_continents

## [1] "Europe" "Americas" "Asia" "Oceania" "Africa" NA
```

Therefore,

$$\frac{100 \text{ countries}}{6 \text{ continents}} \approx 16 \text{ to } 17 \text{ countries per continent}$$

we should randomly sample 17 countries from each continent.

Notably, there are countries that are not on any of the listed continents. Let us see which ones those are:

```
not_in_a_continent = filtered_suicide_df[is.na(filtered_suicide_df$continent),]
write.csv(not_in_a_continent, 'not_in_a_continent.csv')
head(not_in_a_continent)
```

```
##      country year    sex      age suicides_no population continent
## 32317 Rodrigues 2001 female 15-24 years         0         NA      <NA>
## 32318 Rodrigues 2001 female 25-34 years         0         NA      <NA>
## 32319 Rodrigues 2001 female 35-54 years         0         NA      <NA>
## 32320 Rodrigues 2001 female  5-14 years         0         NA      <NA>
## 32321 Rodrigues 2001 female 55-74 years         0         NA      <NA>
## 32322 Rodrigues 2001 female   75+ years         0         NA      <NA>
```

```
unique(not_in_a_continent$country)
```

```
## [1] Rodrigues          Virgin Islands (USA)
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
```

Let us make the choice not to include these countries in the analysis, since there are only two countries.

```
# Take off `NA` from list of continents
list_of_continents <- list_of_continents[-length(list_of_continents)]
list_of_continents
```

```
## [1] "Europe" "Americas" "Asia" "Oceania" "Africa"
```

We will now create six dataframes, filtered by list of countries for each continent.

```
# library(rlist)
countries_per_continent <- list()

for (i in seq_along(list_of_continents))
{
  countries_per_continent[[i]] <- filtered_suicide_df[filtered_suicide_df$continent == list_of_continents[i],]
}
```

```
length(countries_per_continent)
```

```
## [1] 5
```

```
length(countries_per_continent)
```

```
## [1] 5
```

```
for (i in seq_along(countries_per_continent))
{
  print(head(countries_per_continent[[i]]))
  print(length(countries_per_continent[[i]]))
  cat("\n")
}
```

```
##      country year    sex      age suicides_no population continent
## 25 Albania 1987 female 15-24 years         14      289700      Europe
## 26 Albania 1987 female 25-34 years          4      257200      Europe
## 27 Albania 1987 female 35-54 years          6      278800      Europe
## 28 Albania 1987 female  5-14 years          0      311000      Europe
## 29 Albania 1987 female 55-74 years          0      144600      Europe
```

```
## 30 Albania 1987 female 75+ years 1 35600 Europe
## [1] 7
##
## country year sex age suicides_no population continent
## 373 Anguilla 1983 female 15-24 years 0 NA Americas
## 374 Anguilla 1983 female 25-34 years 0 NA Americas
## 375 Anguilla 1983 female 35-54 years 0 NA Americas
## 376 Anguilla 1983 female 5-14 years 0 NA Americas
## 377 Anguilla 1983 female 55-74 years 0 NA Americas
## 378 Anguilla 1983 female 75+ years 0 NA Americas
## [1] 7
##
## country year sex age suicides_no population continent
## 1501 Armenia 1981 female 15-24 years 5 348000 Asia
## 1502 Armenia 1981 female 25-34 years 6 242200 Asia
## 1503 Armenia 1981 female 35-54 years 6 333500 Asia
## 1504 Armenia 1981 female 5-14 years 0 295200 Asia
## 1505 Armenia 1981 female 55-74 years 10 164300 Asia
## 1506 Armenia 1981 female 75+ years 7 43100 Asia
## [1] 7
##
## country year sex age suicides_no population continent
## 2161 Australia 1979 female 15-24 years 71 1236800 Oceania
## 2162 Australia 1979 female 25-34 years 86 1138500 Oceania
## 2163 Australia 1979 female 35-54 years 171 1572100 Oceania
## 2164 Australia 1979 female 5-14 years 1 1246500 Oceania
## 2165 Australia 1979 female 55-74 years 135 1137800 Oceania
## 2166 Australia 1979 female 75+ years 15 309900 Oceania
## [1] 7
##
## country year sex age suicides_no population continent
## 7669 Cabo Verde 2011 female 15-24 years 1 56039 Africa
## 7670 Cabo Verde 2011 female 25-34 years 0 38528 Africa
## 7671 Cabo Verde 2011 female 35-54 years 2 49078 Africa
## 7672 Cabo Verde 2011 female 5-14 years 0 56558 Africa
## 7673 Cabo Verde 2011 female 55-74 years 2 19887 Africa
## 7674 Cabo Verde 2011 female 75+ years 0 7582 Africa
## [1] 7
```

This text links to very important information about why a `for` loop doesn't print anything.¹

[Link to Pandoc Markdown formatting](#)

Randomly sample 17 countries from each continent:

```
list_of_continents
```

```
## [1] "Europe" "Americas" "Asia" "Oceania" "Africa"
for (i in seq_along(countries_per_continent))
{
  print(list_of_continents[i])
  countries <- unique(countries_per_continent[[i]]$country)
```

¹Basically, `for` loops are functions themselves. R prints out the result of a command automatically, but functions are not inherently a command, and since `for` loops are functions, nothing will be printed. The solution is to have `print(command())` within the `for` loop to get output for your `for` loop. You will never again spend hours trying to find out why a `for` loop doesn't print anything because you're no longer an R newbie.

```

print(countries)
print(length(countries))
cat("\n")
}

```

```

## [1] "Europe"
## [1] Albania Austria Belarus
## [4] Belgium Bosnia and Herzegovina Bulgaria
## [7] Croatia Czech Republic Denmark
## [10] Estonia Finland France
## [13] Germany Greece Hungary
## [16] Iceland Ireland Italy
## [19] Latvia Lithuania Luxembourg
## [22] Malta Monaco Montenegro
## [25] Netherlands Norway Poland
## [28] Portugal Republic of Moldova <NA>
## [31] Romania Russian Federation San Marino
## [34] Serbia Slovakia Slovenia
## [37] Spain Sweden Switzerland
## [40] TFYR Macedonia Ukraine United Kingdom
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 42
##
## [1] "Americas"
## [1] Anguilla Antigua and Barbuda
## [3] Argentina Aruba
## [5] Bahamas Barbados
## [7] Belize Bermuda
## [9] Bolivia Brazil
## [11] British Virgin Islands Canada
## [13] Cayman Islands Chile
## [15] Colombia Costa Rica
## [17] Cuba Dominica
## [19] Dominican Republic Ecuador
## [21] El Salvador Falkland Islands (Malvinas)
## [23] French Guiana Grenada
## [25] Guadeloupe Guatemala
## [27] Guyana Haiti
## [29] Honduras Jamaica
## [31] Martinique Mexico
## [33] Montserrat Netherlands Antilles
## [35] Nicaragua Panama
## [37] Paraguay Peru
## [39] Puerto Rico <NA>
## [41] Saint Kitts and Nevis Saint Lucia
## [43] Saint Pierre and Miquelon Saint Vincent and Grenadines
## [45] Suriname Trinidad and Tobago
## [47] Turks and Caicos Islands United States of America
## [49] Uruguay Venezuela (Bolivarian Republic of)
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 50
##
## [1] "Asia"
## [1] Armenia Azerbaijan

```

```
## [3] Bahrain                Brunei Darussalam
## [5] Cyprus                  Georgia
## [7] Hong Kong SAR           Iran (Islamic Rep of)
## [9] Iraq                    Israel
## [11] Japan                   Jordan
## [13] Kazakhstan              Kuwait
## [15] Kyrgyzstan             Macau
## [17] Malaysia                Maldives
## [19] Mongolia                Occupied Palestinian Territory
## [21] Oman                    Philippines
## [23] Qatar                   Republic of Korea
## [25] <NA>                     Saudi Arabia
## [27] Singapore               Sri Lanka
## [29] Syrian Arab Republic    Tajikistan
## [31] Thailand                Turkey
## [33] Turkmenistan            United Arab Emirates
## [35] Uzbekistan
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 35
##
## [1] "Oceania"
## [1] Australia   Fiji           Kiribati     New Zealand <NA>
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 5
##
## [1] "Africa"
## [1] Cabo Verde      Egypt           Mauritius
## [4] Mayotte         Morocco         Reunion
## [7] <NA>            Sao Tome and Principe Seychelles
## [10] South Africa    Tunisia         Zimbabwe
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 12
```

Since there are only 5 countries in Oceania and 12 countries in Africa, we will use all 5 countries of Oceania and all 12 countries of Africa.

```
samples_of_countries <- list()
num_samples <- 17
for (i in seq_along(countries_per_continent))
{
  countries <- unique(countries_per_continent[[i]]$country)
  current_sample <- list()
  if (length(countries) >= num_samples)
  {
    current_sample <- sample(countries, 17)
  } else {
    current_sample <- sample(countries, length(countries))
  }
  samples_of_countries[[i]] <- current_sample
}
```

Let's see the countries that we will be sampling:

```
total <- 0
for (i in seq_along(samples_of_countries))
{
```

```

print(list_of_continents[i])
print(samples_of_countries[[i]])
print(length(samples_of_countries[[i]]))
total <- total + length(samples_of_countries[[i]])
cat("\n")
}

## [1] "Europe"
## [1] Greece          Serbia          Poland
## [4] Austria         Monaco         Germany
## [7] Russian Federation Norway          Switzerland
## [10] Ukraine         Belgium        Bulgaria
## [13] TFYR Macedonia Iceland          Belarus
## [16] Ireland         Portugal
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 17
##
## [1] "Americas"
## [1] Ecuador          Saint Kitts and Nevis
## [3] Cuba             Dominica
## [5] Peru             Netherlands Antilles
## [7] Canada           Dominican Republic
## [9] Bolivia          Martinique
## [11] Cayman Islands   Bermuda
## [13] Chile            Grenada
## [15] Jamaica          Uruguay
## [17] Saint Pierre and Miquelon
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 17
##
## [1] "Asia"
## [1] Tajikistan          Mongolia
## [3] Kyrgyzstan          Armenia
## [5] Occupied Palestinian Territory Hong Kong SAR
## [7] Saudi Arabia        United Arab Emirates
## [9] Syrian Arab Republic Israel
## [11] Jordan              Republic of Korea
## [13] Iraq                Macau
## [15] Iran (Islamic Rep of) Qatar
## [17] Oman
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 17
##
## [1] "Oceania"
## [1] Kiribati <NA> Australia Fiji New Zealand
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe
## [1] 5
##
## [1] "Africa"
## [1] Reunion          South Africa      Zimbabwe
## [4] Egypt           Mauritius         Cabo Verde
## [7] Seychelles      Tunisia           <NA>
## [10] Morocco         Mayotte           Sao Tome and Principe
## 141 Levels: Albania Anguilla Antigua and Barbuda Argentina ... Zimbabwe

```



```
## [1] 12
```

```
total
```

```
## [1] 68
```

Let's filter the original dataframe only to include countries that we have sampled:

```
countries_to_test <- list()
a <- 0
for (i in seq_along(samples_of_countries))
{
  # find out a way to access each country name
  # print each country name
  for (j in seq_along(samples_of_countries[[i]]))
  {
    sample <- samples_of_countries[[i]]
    country_string <- toString(sample[[j]])
    countries_to_test[a] <- country_string
    a <- a + 1
  }
}
```

```
length(countries_to_test)
```

```
## [1] 67
```

```
countries_to_test
```

```
## [[1]]
## [1] "Serbia"
##
## [[2]]
## [1] "Poland"
##
## [[3]]
## [1] "Austria"
##
## [[4]]
## [1] "Monaco"
##
## [[5]]
## [1] "Germany"
##
## [[6]]
## [1] "Russian Federation"
##
## [[7]]
## [1] "Norway"
##
## [[8]]
## [1] "Switzerland"
##
## [[9]]
## [1] "Ukraine"
##
## [[10]]
```

```

## [1] "Belgium"
##
## [[11]]
## [1] "Bulgaria"
##
## [[12]]
## [1] "TFYR Macedonia"
##
## [[13]]
## [1] "Iceland"
##
## [[14]]
## [1] "Belarus"
##
## [[15]]
## [1] "Ireland"
##
## [[16]]
## [1] "Portugal"
##
## [[17]]
## [1] "Ecuador"
##
## [[18]]
## [1] "Saint Kitts and Nevis"
##
## [[19]]
## [1] "Cuba"
##
## [[20]]
## [1] "Dominica"
##
## [[21]]
## [1] "Peru"
##
## [[22]]
## [1] "Netherlands Antilles"
##
## [[23]]
## [1] "Canada"
##
## [[24]]
## [1] "Dominican Republic"
##
## [[25]]
## [1] "Bolivia"
##
## [[26]]
## [1] "Martinique"
##
## [[27]]
## [1] "Cayman Islands"
##
## [[28]]

```

```

## [1] "Bermuda"
##
## [[29]]
## [1] "Chile"
##
## [[30]]
## [1] "Grenada"
##
## [[31]]
## [1] "Jamaica"
##
## [[32]]
## [1] "Uruguay"
##
## [[33]]
## [1] "Saint Pierre and Miquelon"
##
## [[34]]
## [1] "Tajikistan"
##
## [[35]]
## [1] "Mongolia"
##
## [[36]]
## [1] "Kyrgyzstan"
##
## [[37]]
## [1] "Armenia"
##
## [[38]]
## [1] "Occupied Palestinian Territory"
##
## [[39]]
## [1] "Hong Kong SAR"
##
## [[40]]
## [1] "Saudi Arabia"
##
## [[41]]
## [1] "United Arab Emirates"
##
## [[42]]
## [1] "Syrian Arab Republic"
##
## [[43]]
## [1] "Israel"
##
## [[44]]
## [1] "Jordan"
##
## [[45]]
## [1] "Republic of Korea"
##
## [[46]]

```

```

## [1] "Iraq"
##
## [[47]]
## [1] "Macau"
##
## [[48]]
## [1] "Iran (Islamic Rep of)"
##
## [[49]]
## [1] "Qatar"
##
## [[50]]
## [1] "Oman"
##
## [[51]]
## [1] "Kiribati"
##
## [[52]]
## [1] "NA"
##
## [[53]]
## [1] "Australia"
##
## [[54]]
## [1] "Fiji"
##
## [[55]]
## [1] "New Zealand"
##
## [[56]]
## [1] "Reunion"
##
## [[57]]
## [1] "South Africa"
##
## [[58]]
## [1] "Zimbabwe"
##
## [[59]]
## [1] "Egypt"
##
## [[60]]
## [1] "Mauritius"
##
## [[61]]
## [1] "Cabo Verde"
##
## [[62]]
## [1] "Seychelles"
##
## [[63]]
## [1] "Tunisia"
##
## [[64]]

```

```
## [1] "NA"  
##  
## [[65]]  
## [1] "Morocco"  
##  
## [[66]]  
## [1] "Mayotte"  
##  
## [[67]]  
## [1] "Sao Tome and Principe"
```

4. Interpretation of the results or discussion

5. References

Efron, Bradley. 2003. *Second Thoughts on the Bootstrap*. Department of Biostatistics, Stanford University.

Efron, Bradley, and Robert Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman; Hall.