Fundamentals of Computing and Data Display

Assignment 2

Flavia Batista da Silva

Setup

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

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

```
library(gtrendsR)
```

```
## Warning: package 'gtrendsR' was built under R version 4.1.2
```

```
library(censusapi)
```

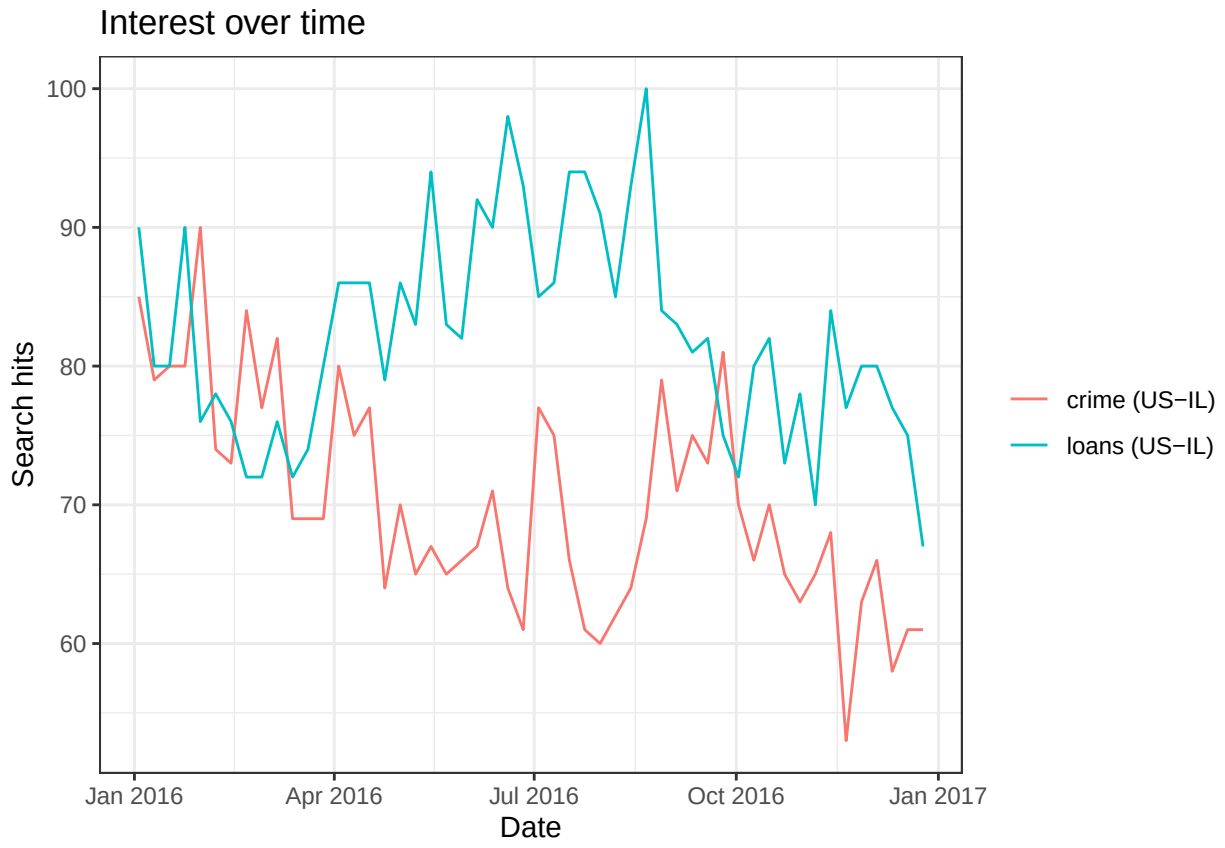
```
## Warning: package 'censusapi' was built under R version 4.1.2

##
## Attaching package: 'censusapi'

## The following object is masked from 'package:methods':
##
##     getFunction
```

Google Trends

In this notebook, your task is to combine and explore web data using APIs and `dplyr`. Try to utilize piping in this notebook when writing your code.

Our first data source is the Google Trends API. This time we are interested in the search trends for `crime` and `loans` in Illinois in the year 2016.

```
res <- gtrends(c("crime", "loans"), geo = "US-IL", time = "2016-01-01 2016-12-31", low_search_volume = T
plot(res)
```



The resulting list includes a `data.frame` with the search interest by city. Extract this data set as a `tibble` and print the first few observations.

```
interest_city <-as_tibble(res$interest_by_city) #extract as tibble
head(interest_city) #print the first few observations
```

```
## # A tibble: 6 x 5
##   location     hits keyword geo   gprop
##   <chr>       <int> <chr>   <chr> <chr>
## 1 Riverwoods    100 crime   US-IL web
## 2 Canton         55 crime   US-IL web
## 3 Vandalia       55 crime   US-IL web
## 4 Palos Park     40 crime   US-IL web
## 5 Riverdale      31 crime   US-IL web
## 6 River Grove    30 crime   US-IL web
```

Find the mean, median and variance of the search hits for the keywords `crime` and `loans`. This can be done via piping with `dplyr`.

```
interest_city %>%
  group_by(keyword) %>%
  summarize(mean = mean(hits, na.rm = T), median = median(hits, na.rm = T), variance = var(hits, na.rm =
```

```
## # A tibble: 2 x 4
##   keyword  mean median variance
##   <chr>   <dbl>  <dbl>    <dbl>
## 1 crime    22.7   22       139.
## 2 loans    17.9   13.5     240.
```

Note that there might be multiple rows for each city if there were hits for both "crime" and "loans" in that city. It might be easier if we had the search hits info for both search terms in two separate variables. That is, each row would represent a unique city. Transform the `tibble` accordingly and save the result as a new object.

```
interest_city_w <- pivot_wider(interest_city,
                           names_from = keyword,
                           values_from = hits)
interest_city_w
```

```
## # A tibble: 359 x 5
##    location    geo   gprop crime loans
##    <chr>       <chr> <chr> <int> <int>
##  1 Riverwoods  US-IL web     100    NA
##  2 Canton      US-IL web      55    NA
##  3 Vandalia    US-IL web      55    17
##  4 Palos Park  US-IL web      40    NA
##  5 Riverdale   US-IL web      31    13
##  6 River Grove US-IL web      30    NA
##  7 Wayne       US-IL web      30    NA
##  8 Macomb      US-IL web      29    NA
##  9 Lebanon     US-IL web      28    NA
## 10 Palos Hills US-IL web      28    NA
## # ... with 349 more rows
```

Which cities (locations) have the highest search frequency for `loans`? Print the first rows of the new `tibble` from the previous chunk, ordered by `loans`.

```
interest_city_w %>%
  arrange(desc(loans))
```

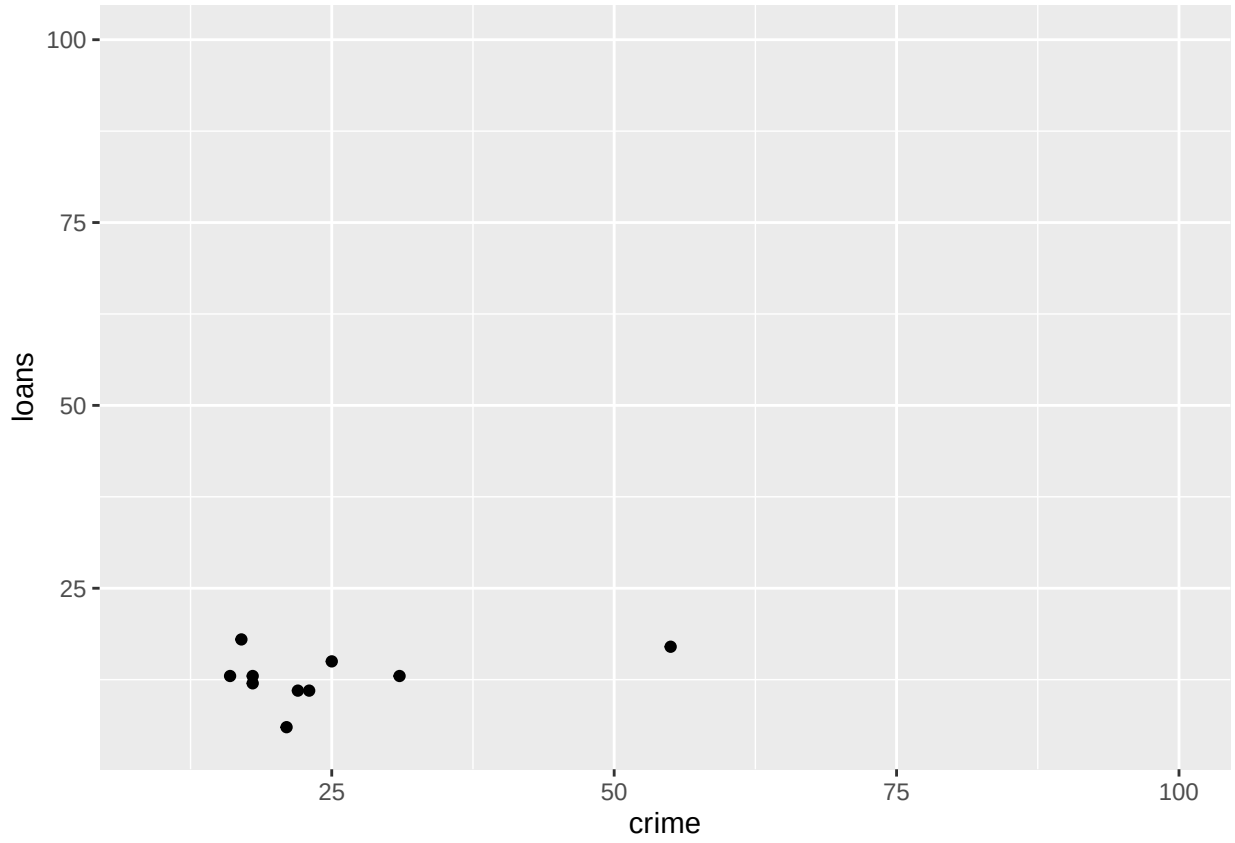
```
## # A tibble: 359 x 5
##    location  geo   gprop crime loans
##    <chr>     <chr> <chr> <int> <int>
##  1 Coffeen   US-IL web      NA   100
##  2 Palestine US-IL web      NA    85
##  3 Warsaw    US-IL web      NA    60
##  4 Hanover   US-IL web      NA    43
##  5 Robbins   US-IL web      NA    40
##  6 Fruitland US-IL web      NA    34
##  7 Durand    US-IL web      NA    33
##  8 Mapleton  US-IL web      NA    33
##  9 Savanna   US-IL web      NA    29
## 10 Zion      US-IL web      NA    28
## # ... with 349 more rows
```

```
#Riverton, Zion, Madison, and Robbin are the locations with the highest seach frequency for "loans".
```

Is there a relationship between the search intensities between the two keywords we used? Create a scatterplot of `crime` and `loans` with `qplot()`.

```
#There seems not to be a relationship between the search intensities of "loans" and "crime".

qplot(crime, loans, data = interest_city_w)
```

```
## Warning: Removed 349 rows containing missing values (geom_point).
```



Google Trends + ACS

Now lets add another data set. The censusapi package provides a nice R interface for communicating with this API. However, before running queries we need an access key. This (easy) process can be completed here:

https://api.census.gov/data/key_signup.html

Once you have an access key, store this key in the cs_key object. We will use this object in all following API queries.

```
cs_key <- "f2f207f9e8d30faf836a37450b39f4054c8b395a"
```

In the following, we request basic socio-demographic information (population, median age, median household income, income per capita) for cities and villages in the state of Illinois.

```
acs_il <- getCensus(name = "acs/acs5",
                    vintage = 2016,
                    vars = c("NAME", "B01001_001E", "B06002_001E", "B19013_001E", "B19301_001E"),
                    region = "place:*",
                    regionin = "state:17",
                    key = cs_key)
head(acs_il)
```

```
##   state place                           NAME B01001_001E B06002_001E B19013_001E
## 1    17 11202    Carlinville city, Illinois        5297        36.7       40250
## 2    17 21410  Eagarville village, Illinois         165        39.2       48750
## 3    17 57043     Owaneco village, Illinois         201        44.6       42500
## 4    17 34137     Henning village, Illinois         243        31.9       55500
## 5    17 00880    Allerton village, Illinois         288        42.6       58125
## 6    17 57693 Parkersburg village, Illinois         146        41.1       48000
##   B19301_001E
## 1       22441
## 2       31400
## 3       22708
## 4       18009
## 5       24356
## 6       24795
```

Convert values that represent missings to NAs.

```
acs_il[acs_il == -666666666] <- NA
```

Now, it might be useful to rename the socio-demographic variables (`B01001_001E` etc.) in our data set and assign more meaningful names.

```
acs_il <-
  acs_il %>%
  rename(pop = B01001_001E, age = B06002_001E, hh_income = B19013_001E, income = B19301_001E)
```

Print the first rows of the variable `NAME`.

```
head(acs_il$NAME)
```

```
## [1] "Carlinville city, Illinois"    "Eagarville village, Illinois"
## [3] "Owaneco village, Illinois"     "Henning village, Illinois"
## [5] "Allerton village, Illinois"    "Parkersburg village, Illinois"
```

It seems like we could try to use this location information listed above to merge this data set with the Google Trends data. However, we first have to clean `NAME` so that it has the same structure as `location` in the search interest by city data. Add a new variable `location` to the ACS data that only includes city names.

```
acs_il2 <- acs_il %>% separate(NAME, c('location', 'NAME'))
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 1368 rows [1, 2, 3,
## 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

First, check how many cities don't appear in both data sets, i.e. cannot be matched.

That's a lot, unfortunately. However, we can still try using the data. Create a new data set by joining the Google Trends and the ACS data. Keep only cities that appear in both data sets.

```
merged_data<- inner_join(interest_city_w, acs_il2, by ='location')
head(merged_data)
```

```
## # A tibble: 6 x 12
##   location   geo   gprop crime loans state place NAME      pop   age hh_income
##   <chr>      <chr> <chr> <int> <int> <chr> <chr> <chr>   <dbl> <dbl>     <dbl>
## 1 Riverwoods US-IL web     100    NA 17    64538 village  3759  48.3    187857
## 2 Canton     US-IL web      55    NA 17    11007 city    14397  39.6     39248
## 3 Vandalia   US-IL web      55    17 17    77317 city     6758  36.5     44455
## 4 Riverdale  US-IL web      31    13 17    64278 village 13047  35       31438
## 5 Wayne      US-IL web      30    NA 17    79436 City      951  45.8     45571
## 6 Wayne      US-IL web      30    NA 17    79397 village  2513  49.1    145875
## # ... with 1 more variable: income <dbl>
```

Now we can utilize information from both data sources. As an example, print the `crime` and `loans` search popularity for the first ten cities in Illinois with the highest population (in 2016).

```
merged_data %>%
  slice_max(pop, n = 10)
```

```
## # A tibble: 10 x 12
##    location    geo   gprop crime loans state place NAME       pop   age hh_income
##    <chr>       <chr> <chr> <int> <int> <chr> <chr> <chr>    <dbl> <dbl>     <dbl>
##  1 Rockford    US-IL web      22    NA 17    65000 city    149597  36       40143
##  2 Bloomington US-IL web      18    NA 17    06613 city     78368  34.4     63115
##  3 Evanston    US-IL web      22    NA 17    24582 city     75472  35.3     71317
##  4 Normal      US-IL web      17    NA 17    53234 town     54534  23.9     54496
##  5 Hoffman     US-IL web      NA    NA 17    35411 Estates  51727  37.8     88733
##  6 Bartlett    US-IL web      20    NA 17    04013 village  41475  38.9    100458
##  7 Hanover     US-IL web      NA    43 17    32746 Park     38331  33.9     69922
##  8 Lansing     US-IL web      23    NA 17    42028 village  28369  40.9     50107
##  9 Alton       US-IL web      NA    11 17    01114 city     27175  37.5     37108
## 10 Vernon      US-IL web      NA    NA 17    77694 Hills    25910  38.4     95217
## # ... with 1 more variable: income <dbl>
```

Next, compute the mean of the search popularity for both keywords for cities that have an above average median household income and for those that have an below average median household income. When building your pipe, start with creating the grouping variable and then proceed with the remaining tasks.

```
merged_data$hh_income_level <- "Below hh income"

merged_data$hh_income_level[which(merged_data$hh_income > 47625)] <- "Above hh income"

mean_crime_hhincome <- merged_data %>%
group_by(hh_income_level) %>%
summarise(mean_crime = mean(crime, na.rm = TRUE))

mean_loans_hhincome <- merged_data %>%
```

```
group_by(hh_income_level) %>%
summarise(mean_loans = mean(loans, na.rm = TRUE))

mean_crime_hhincome
```

```
## # A tibble: 2 x 2
##   hh_income_level mean_crime
##   <chr>                <dbl>
## 1 Above hh income       22.6
## 2 Below hh income       24.3
```

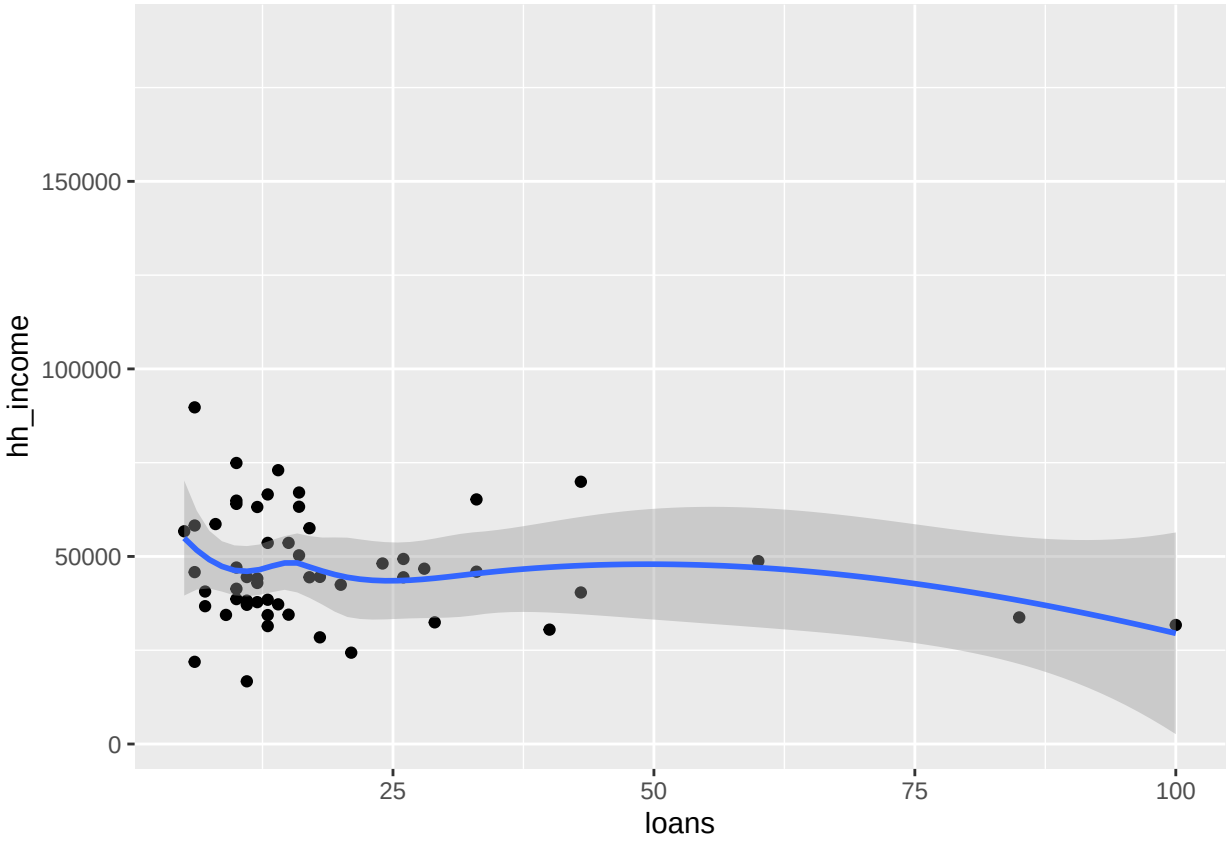
```
mean_loans_hhincome
```

```
## # A tibble: 2 x 2
##   hh_income_level mean_loans
##   <chr>                <dbl>
## 1 Above hh income       17.8
## 2 Below hh income       20.9
```

Is there a relationship between the median household income and the search popularity of `loans`? Plot a scatterplot with `qplot()`.

```
#There seems to be a  positive but weak relationship between household income and the search popularity

qplot(loans, hh_income, data=merged_data, , geom = c("point", "smooth"))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 219 rows containing non-finite values (stat_smooth).

## Warning: Removed 219 rows containing missing values (geom_point).
```


150000
100000
50000
0
hh_income
25
50
75
100
loans