Assignment 02

2333066 白玉龙

2023-11-13

Part 1: Using the iri dateset

Step 1: Get a subset with STATE_CODE 6 and SHRP_ID starting with 050.

```
data_02 <- read.csv("../Data/LTPP/iri.csv")
sub_data <- data_02 |>
            dplyr::filter(STATE_CODE==6, grepl("050", SHRP_ID))

sub_data |>
  head(10)
```

```
##    STATE_CODE SHRP_ID CONSTRUCTION_NO           VISIT_DATE    IRI
## 1           6    0501               1 1/25/90, 12:00:00 AM 3.7206
## 2           6    0501               1 2/11/92, 12:00:00 AM 3.8896
## 3           6    0501               1 2/16/91, 12:00:00 AM 3.6434
## 4           6    0501               2 2/11/98, 12:00:00 AM 1.4376
## 5           6    0501               2  2/2/93, 12:00:00 AM 1.1230
## 6           6    0501               2 2/27/97, 12:00:00 AM 1.3510
## 7           6    0501               2  3/5/99, 12:00:00 AM 1.7332
## 8           6    0501               2  4/4/95, 12:00:00 AM 1.2378
## 9           6    0501               3 3/10/00, 12:00:00 AM 1.9154
## 10          6    0501               4 2/12/02, 12:00:00 AM 1.6550
```

Step 2: obtain the summary statistics of IRI of each section: min, max, and mean

```
# drop NA -  also use drop.na()
data_02_noIRINA <-  dplyr::filter(data_02, !is.na(IRI))

summary_stat <- data_02_noIRINA |>
                dplyr::group_by(STATE_CODE, SHRP_ID) |>
                summarise(
                  min_value = min(IRI),
```

```
                  max_value = max(IRI),
                  mean_value = mean(IRI),
                  .groups = 'drop'
)

summary_stat |>
  head(10)
```

```
## # A tibble: 10 x 5
##    STATE_CODE SHRP_ID min_value max_value mean_value
##         <int> <chr>       <dbl>     <dbl>      <dbl>
##  1          1 0101        0.657     0.810      0.716
##  2          1 0102        0.897     3.10       1.34
##  3          1 0103        0.760     0.834      0.803
##  4          1 0104        0.594     0.684      0.644
##  5          1 0105        0.614     0.694      0.648
##  6          1 0106        0.582     0.764      0.688
##  7          1 0107        0.628     0.963      0.747
##  8          1 0108        0.731     0.875      0.766
##  9          1 0109        0.679     0.775      0.736
## 10          1 0110        0.672     0.764      0.705
```

Step 3: Sort the summarized data by the averaged IRI in a descending order (report results for one section only)

注意: 此处, 我们将 iri.csv 数据集中的 IRI 当作 Averaged IRI 进行展示

```
# 此处只展示 STATE_CODE 等于 6 且 SHRP_ID 等于 0501 这个 section 的结果
any_section <- data_02_noIRINA |>
               dplyr::filter(STATE_CODE==6, grepl("0501", SHRP_ID))

any_section |>
  dplyr::arrange(desc(IRI))
```

```
##    STATE_CODE SHRP_ID CONSTRUCTION_NO           VISIT_DATE    IRI
## 1           6    0501               1 2/11/92, 12:00:00 AM 3.8896
## 2           6    0501               1 1/25/90, 12:00:00 AM 3.7206
## 3           6    0501               1 2/16/91, 12:00:00 AM 3.6434
## 4           6    0501               5 3/20/07, 12:00:00 AM 3.0780
## 5           6    0501               4 3/12/05, 12:00:00 AM 2.4044
## 6           6    0501               4 3/25/04, 12:00:00 AM 1.9188
## 7           6    0501               3 3/10/00, 12:00:00 AM 1.9154
```

```
## 8           6    0501              2  3/5/99, 12:00:00 AM 1.7332
## 9           6    0501              4 2/12/02, 12:00:00 AM 1.6550
## 10          6    0501              4  3/5/03, 12:00:00 AM 1.5458
## 11          6    0501              2 2/11/98, 12:00:00 AM 1.4376
## 12          6    0501              4 2/17/01, 12:00:00 AM 1.4026
## 13          6    0501              2 2/27/97, 12:00:00 AM 1.3510
## 14          6    0501              2  4/4/95, 12:00:00 AM 1.2378
## 15          6    0501              2  2/2/93, 12:00:00 AM 1.1230
```

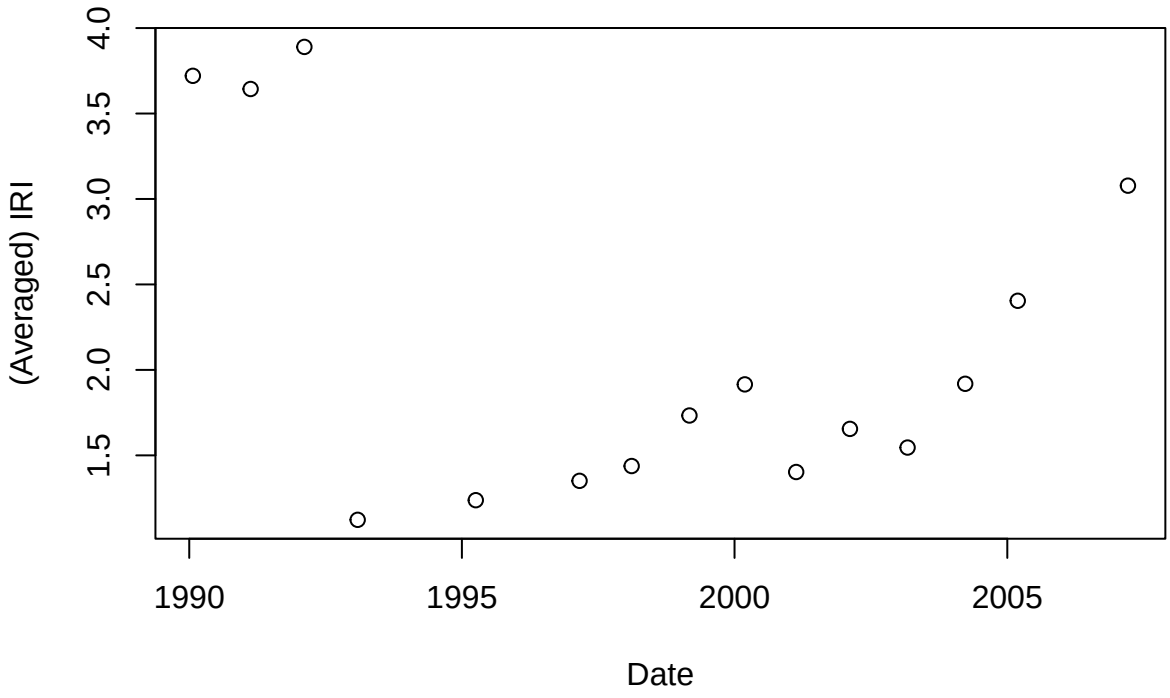
Step 4: Generate a scatter plot for the averaged IRI against the time for a selected section, and then give your interpretation of the plot:

- *HINT 1: mean IRI vs. date*
- *HINT 2: STATE_CODE and SHRP_ID together to form a primary key that uniquely identifies a section.*

注意: 此处，我们仍然将 iri.csv 数据集中的 IRI 当作 Averaged IRI 进行处理

```r
# transform the str {%y-%m-%d} into the timestamp
str2stamp <- function(object) {
             data.frame(DATE_FIX = strptime(object, format = "%y-%m-%d" ))
}

# extract the {%y-%m-%d} from the origin date
date_fix <- data.frame()
for (i in any_section$VISIT_DATE) {
  month_str <- strsplit(i, "/")[[1]][1]
  day_str <- strsplit(i, "/")[[1]][2]
  mid_str <- strsplit(i, "/")[[1]][3]
  year_str <- strsplit(mid_str, ",")[[1]][1]

  tstmap <- paste(year_str, month_str, day_str, sep = "-") |>
            str2stamp()

  date_fix <- rbind(date_fix, tstmap)
}

# plot data
plot(
  x = date_fix$DATE_FIX,
  y = any_section$IRI,
  xlab = "Date",
```

```
  ylab = "(Averaged) IRI"
)
```



Interpretation : 从绘图中可以看出，除了 1990 年的（平均）IRI 过高，从 1995 年依赖到 2005 年的（平均）IRI 在逐渐升高。这种现象有可能是因为 IRI 的计算方式不一致导致的。

Part 2: Using the CRSS datasets in 2017

Step 1: Get the intersection of the datasets accident and person

```
accidents <- read.csv("../Data/CRSS/ACCIDENT.csv")
persons <- read.csv("../Data/CRSS/PERSON.csv")
vehicles <- read.csv("../Data/CRSS/VEHICLE.CSV")

acc_per_cross <- inner_join(
                 x = accidents,
                 y = persons,
                 by = c("CASENUM", "PSU")) |>
                 distinct()

acc_per_cross |>
```

```
  head(1)
```

```
##     CASENUM REGION.x PSU PJ.x PSU_VAR.x URBANICITY.x STRATUM.x VE_TOTAL
## 1 2.017e+11        4  64  305        64            1         3        2
##   VE_FORMS.x PVH_INVL PEDS PERMVIT PERNOTMVIT NUM_INJ MONTH.x YEAR DAY_WEEK
## 1          2        0    0       2          0       1       3 2017        4
##   HOUR.x MINUTE.x HARM_EV.x ALCOHOL MAX_SEV MAN_COLL.x RELJCT1 RELJCT2 TYP_INT
## 1     17       45        12       2       2          6       8       2       6
##   WRK_ZONE REL_ROAD LGT_COND WEATHER1 WEATHER2 WEATHER SCH_BUS.x INT_HWY CF1
## 1        0        1        1        1        0       1         0       0   0
##   CF2 CF3 WKDY_IM HOUR_IM MINUTE_IM EVENT1_IM MANCOL_IM RELJCT1_IM RELJCT2_IM
## 1   0   0       4      17        45        12         6          0          2
##   LGTCON_IM WEATHR_IM MAXSEV_IM NO_INJ_IM ALCHL_IM PSUSTRAT.x WEIGHT.x
## 1         1         1         2         1        2         23 29.27031
##   VE_FORMS.y VEH_NO PER_NO REGION.y PJ.y PSU_VAR.y URBANICITY.y STRATUM.y
## 1          2      1      1        4  305        64            1         3
##   STR_VEH MONTH.y HOUR.y MINUTE.y HARM_EV.y MAN_COLL.y SCH_BUS.y MAKE BODY_TYP
## 1       0       3     17       45        12          6         0   35        9
##   MOD_YEAR MAK_MOD TOW_VEH SPEC_USE EMER_USE ROLLOVER IMPACT1 FIRE_EXP AGE SEX
## 1     1995   35032       0        0        0        0      62        0  51   2
##   PER_TYP INJ_SEV SEAT_POS REST_USE REST_MIS AIR_BAG EJECTION DRINKING
## 1       1       0       11        3        0      20        7        0
##   ALC_STATUS ATST_TYP ALC_RES DRUGS DSTATUS DRUGTST1 DRUGTST2 DRUGTST3 DRUGRES1
## 1          8       95     995     0       8        6        0        0       95
##   DRUGRES2 DRUGRES3 HOSPITAL P_SF1 P_SF2 P_SF3 LOCATION SEX_IM INJSEV_IM
## 1        0        0        0     0     0     0        0      2         0
##   EJECT_IM PERALCH_IM SEAT_IM AGE_IM PSUSTRAT.y WEIGHT.y
## 1        0          0      11     51         23 29.27031
```

Step 2: Tabulate the total number of observations in each injury severity (INJ_SEV)

- *HINT: use summarise() and group_by()*

查找原数据可得，*MAX_SEV* 是事故类型，*NUM_TNJ* 是伤亡人数

https://www.nhtsa.gov/file-downloads?p=nhtsa/downloads/CRSS/

```
acc_view <- accidents|>
            group_by(MAX_SEV)|>
            summarise(
              INJ_SEV = sum(NUM_INJ),
              .groups = 'drop'
            )
```

```r
acc_view |>
  head(10)
```

```
## # A tibble: 9 x 2
##   MAX_SEV INJ_SEV
##     <int>   <int>
## 1       0       0
## 2       1   17688
## 3       2   12258
## 4       3   10119
## 5       4    1903
## 6       5     419
## 7       6       0
## 8       8    1960
## 9       9   97911
```

Step 3: Merge the accident dataset with the vehicle dataset, and report the dimension of your results and number of missing values in one variable of the right dataset

- *HINT: left_join()*

```r
acc_veh_left <- left_join(
                x = accidents,
                y = vehicles,
                by = c("CASENUM", "PSU")) |>
                distinct()

acc_veh_left_dim <- dim(acc_veh_left)
num_of_missing <- colSums(is.na(acc_veh_left))

cat("The dimension of my result: ", acc_veh_left_dim, "\n")
cat("The number of missing values in all variables of the right dataset: ",
     num_of_missing,
     fill = TRUE
)
## The dimension of my result:  97625 136
## The number of missing values in all variables of the right dataset:  0 0 0 0 0
## 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
## 0 0 0 0 0 0 0 0 0 0 0 0
```

Analysis：根据最后的结果，我们可以得出结论，在将事故数据集与车辆数据集合并时，车辆数据集中的所有变量都没有缺失值。