STAT 612 Week 7 Homework_Messy

Data readr and tidyr

Yunting

2/29/2020

```
suppressMessages(library(tidyverse))
```

Exercises

1. Baltimore City Crime Data:

a. Import the data from https://dcgerard.github.io/stat_412_612/data/BPD_Part_1_Victim_Based_Crime_Data.zip.

```
Baltimore_crime <- read_csv(file = "./data/BPD_Part_1_Victim_Based_Crime_Data.csv")
```

```
## Parsed with column specification:
## cols(
##   CrimeDate = col_character(),
##   CrimeTime = col_character(),
##   CrimeCode = col_character(),
##   Location = col_character(),
##   Description = col_character(),
##   `Inside/Outside` = col_character(),
##   Weapon = col_character(),
##   Post = col_double(),
##   District = col_character(),
##   Neighborhood = col_character(),
##   Longitude = col_double(),
##   Latitude = col_double(),
##   `Location 1` = col_character(),
##   Premise = col_character(),
##   crimeCaseNumber = col_logical(),
##   `Total Incidents` = col_double()
## )
```

```
head(Baltimore_crime)
```

```
## # A tibble: 6 x 16
##   CrimeDate CrimeTime CrimeCode Location Description `Inside/Outside` Weapon
##   <chr>     <chr>     <chr>     <chr>    <chr>       <chr>            <chr>
## 1 12/08/20... 23:20:00  4E        100 S E... COMMON ASS... I                <NA>
## 2 12/08/20... 23:00:00  6D        900 S C... LARCENY FR... O                <NA>
## 3 12/08/20... 23:00:00  6D        2600 HU... LARCENY FR... O                <NA>
## 4 12/08/20... 22:50:00  7A        3800 MA... AUTO THEFT  O                <NA>
## 5 12/08/20... 22:49:00  4E        300 S C... COMMON ASS... I                <NA>
## 6 12/08/20... 22:15:00  3AF       NORTH A... ROBBERY - ... O                FIREA...
## # ... with 9 more variables: Post <dbl>, District <chr>, Neighborhood <chr>,
## #   Longitude <dbl>, Latitude <dbl>, `Location 1` <chr>, Premise <chr>,
## #   crimeCaseNumber <lgl>, `Total Incidents` <dbl>
```

b. Convert the given dates and times to date classes. For CrimeTime, not all of the rows conform to the "HH:MM:SS" format. I'll give you a point extra credit if you successfuly demonstrate you fixed all of those locations.

```
# cauculate the length of CrimeTime with each type.
Baltimore_crime %>%
  select(CrimeTime) %>%
  mutate(length_crimeTime = str_length(CrimeTime)) %>%
  #str_length: calculate the length of number length
  group_by(length_crimeTime) %>%
  summarise(count = length(length_crimeTime))
```

```
## # A tibble: 8 x 2
##   length_crimeTime  count
##              <int>  <int>
## 1                2      2
## 2                3      6
## 3                4   5738
## 4                5      4
## 5                7      4
## 6                8 338616
## 7               10      1
## 8               NA     16
```

```
# select the CrimeTime of each row looks like
Baltimore_crime %>%
   filter(str_length(CrimeTime) == 7 )
```

```
## # A tibble: 4 x 16
##   CrimeDate CrimeTime CrimeCode Location Description `Inside/Outside` Weapon
##   <chr>     <chr>     <chr>     <chr>    <chr>       <chr>            <chr>
## 1 01/05/20... 2:51:00   9S        6600 HA... SHOOTING    <NA>             FIREA...
## 2 01/05/20... 1:45:00   9S        2700 W ... SHOOTING    <NA>             FIREA...
## 3 12/30/20... 5:32:00   1F        400 GOL... HOMICIDE    <NA>             FIREA...
## 4 01/26/20... 4:25:00   1K        2500 E ... HOMICIDE    <NA>             KNIFE
## # ... with 9 more variables: Post <dbl>, District <chr>, Neighborhood <chr>,
## #   Longitude <dbl>, Latitude <dbl>, `Location 1` <chr>, Premise <chr>,
## #   crimeCaseNumber <lgl>, `Total Incidents` <dbl>
```

```r
# Let's do it!!

table_2 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 2) %>%
  mutate(CrimeTime = parse_time(CrimeTime, format = "%H"))
         #%H default all zero after Hours

table_3 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 3) %>%
  mutate(CrimeTime = paste("0",CrimeTime, sep=""),
#sep: combined two kinds of stuff with "", can put anything inside to be a glue
         CrimeTime = parse_time(CrimeTime, format = "%H%M" ))

table_4 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 4) %>%
  mutate(CrimeTime = recode(CrimeTime, "2400" = "0000"),
  # or: mutate(CrimeTime = if_else(CrimeTime == "2400" , "0000", CrimeTime ),
    CrimeTime = parse_time(CrimeTime, format = "%H%M"))

table_5 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 5) %>%
  mutate(CrimeTime = if_else(str_detect(CrimeTime, ":"),
                             CrimeTime,
                             as.character(parse_number(CrimeTime))) ,
         CrimeTime =  if_else(str_detect(CrimeTime, ":"),
                             CrimeTime,
                             str_c(str_sub(CrimeTime,1,2),
                                   str_sub(CrimeTime,3,), sep = ":")),
         CrimeTime = parse_time(CrimeTime, format = "%H:%M"))

table_7 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 7) %>%
 mutate(CrimeTime = parse_time(CrimeTime, format = "%H:%M:%S"))

table_8 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 8) %>%
  mutate(CrimeTime = parse_time(CrimeTime, format = "%H:%M:%S" ))

table_10 <- Baltimore_crime %>%
  filter(str_length(CrimeTime) == 10) %>%
  mutate(CrimeTime = str_sub("0149 01:49",5),
         # or str_sub("0149 01:49",5,10)
    CrimeTime = parse_time(CrimeTime, format = "%H:%M" ))
    #%H:%M default all zero after Minutes

table_NA <- Baltimore_crime %>%
  filter(is.na(CrimeTime)) %>%
  mutate(CrimeTime = parse_time(CrimeTime, format = "%H:%M:%S" ))

Baltimore_crime_data_exhausted <- table_2 %>%
  full_join(table_3) %>%
  full_join(table_4) %>%
  full_join(table_5) %>%
  full_join(table_7) %>%
  full_join(table_8) %>%
  full_join(table_10) %>%
  full_join(table_NA)
```

```
## Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")Joining, by = c("CrimeDate", "CrimeTime", "CrimeCode", "Location",
## "Description", "Inside/Outside", "Weapon", "Post", "District", "Neighborhood",
## "Longitude", "Latitude", "Location 1", "Premise", "crimeCaseNumber", "Total
## Incidents")
```

```
# or: rbind(table_2,table_3,table_4,table_5,
# table_7,table_8,table_10,table_NA)
# use full_join needs to have same format, including "table_NA"

Baltimore_crime_data_exhausted
```

```
## # A tibble: 344,387 x 16
##    CrimeDate CrimeTime CrimeCode Location Description `Inside/Outside` Weapon
##    <chr>     <time>    <chr>     <chr>    <chr>       <chr>            <chr>
##  1 10/11/20... 10:00     9S        800 N G... SHOOTING    O                FIREA...
##  2 01/01/20... 10:00     9S        600 LIG... SHOOTING    <NA>             FIREA...
##  3 12/05/20... 02:42     9S        3000 S ... SHOOTING    Outside          FIREA...
##  4 01/04/20... 08:34     9S        300 N H... SHOOTING    <NA>             FIREA...
##  5 01/04/20... 02:56     9S        200 E 2... SHOOTING    <NA>             FIREA...
##  6 01/04/20... 01:07     9S        4000 GA... SHOOTING    <NA>             FIREA...
##  7 01/03/20... 02:21     9S        2400 W ... SHOOTING    <NA>             FIREA...
##  8 01/03/20... 01:15     9S        5300 FR... SHOOTING    <NA>             FIREA...
##  9 12/08/20... 04:54     9S        4300 GA... SHOOTING    I                FIREA...
## 10 12/07/20... 22:47     1F        4700 N ... HOMICIDE    O                FIREA...
## # ... with 344,377 more rows, and 9 more variables: Post <dbl>, District <chr>,
## #   Neighborhood <chr>, Longitude <dbl>, Latitude <dbl>, `Location 1` <chr>,
## #   Premise <chr>, crimeCaseNumber <lgl>, `Total Incidents` <dbl>
```

```
#Test area
#Baltimore_crime %>%
#filter(str_length(CrimeTime) == 4, CrimeTime == "2400")  %>%
#mutate(CrimeTime = if_else(CrimeTime == "2400" , "0000", CrimeTime ))
```

If you cannot figure it out, remove those rows where the parsing failed.

###c.Make Location 1 into two columns LocationLat and LocationLon

```
Baltimore_crime_data_exhausted  %>%
 separate("Location 1", into = c("LocationLat", "LocationLon"), sep = ",") %>%
  mutate(LocationLat = parse_number(LocationLat),
         LocationLon = parse_number(LocationLon)) %>%
  head()
```

```
## # A tibble: 6 x 17
##   CrimeDate CrimeTime CrimeCode Location Description `Inside/Outside` Weapon
##   <chr>     <time>    <chr>     <chr>    <chr>       <chr>            <chr>
## 1 10/11/20... 10:00     9S        800 N G... SHOOTING    0                FIREA...
## 2 01/01/20... 10:00     9S        600 LIG... SHOOTING    <NA>             FIREA...
## 3 12/05/20... 02:42     9S        3000 S ... SHOOTING    Outside          FIREA...
## 4 01/04/20... 08:34     9S        300 N H... SHOOTING    <NA>             FIREA...
## 5 01/04/20... 02:56     9S        200 E 2... SHOOTING    <NA>             FIREA...
## 6 01/04/20... 01:07     9S        4000 GA... SHOOTING    <NA>             FIREA...
## # ... with 10 more variables: Post <dbl>, District <chr>, Neighborhood <chr>,
## #   Longitude <dbl>, Latitude <dbl>, LocationLat <dbl>, LocationLon <dbl>,
## #   Premise <chr>, crimeCaseNumber <lgl>, `Total Incidents` <dbl>
```

d. Determine the % of crimes committed between midnight and 4:00 am.

```
Midnight_to_4 <- Baltimore_crime_data_exhausted %>%
  filter(CrimeTime >= parse_time("00:00:00", format = "%H:%M:%S") &
         CrimeTime <= parse_time("04:00:00", format = "%H:%M:%S"))

percentage_of_midnight <- (nrow(Midnight_to_4)/
                             nrow(Baltimore_crime_data_exhausted))*100
percentage_of_midnight
```

```
## [1] 14.03392
```

2. Import the billboard dataset (posted as a .csv on Blackboard) and tidy it up. The values in column *wkx* are a song's ranking after x weeks of being released.

```
billboard <- read_csv(file = "./data/billboard.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   artist = col_character(),
##   track = col_character(),
##   time = col_time(format = ""),
##   date.entered = col_date(format = ""),
##   wk66 = col_logical(),
##   wk67 = col_logical(),
##   wk68 = col_logical(),
##   wk69 = col_logical(),
##   wk70 = col_logical(),
##   wk71 = col_logical(),
##   wk72 = col_logical(),
##   wk73 = col_logical(),
##   wk74 = col_logical(),
##   wk75 = col_logical(),
##   wk76 = col_logical()
## )
```

```
## See spec(...) for full column specifications.
```

```
head(billboard)
```

```
## # A tibble: 6 x 81
##    year artist track time  date.entered   wk1   wk2   wk3   wk4   wk5   wk6
##   <dbl> <chr>  <chr> <tim> <date>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  2000 2 Pac  Baby... 04:22 2000-02-26     87    82    72    77    87    94
## 2  2000 2Ge+h... The ... 03:15 2000-09-02     91    87    92    NA    NA    NA
## 3  2000 3 Doo... Kryp... 03:53 2000-04-08     81    70    68    67    66    57
## 4  2000 3 Doo... Loser 04:24 2000-10-21     76    76    72    69    67    65
## 5  2000 504 B... Wobb... 03:35 2000-04-15     57    34    25    17    17    31
## 6  2000 98^0   Give... 03:24 2000-08-19     51    39    34    26    26    19
## # ... with 70 more variables: wk7 <dbl>, wk8 <dbl>, wk9 <dbl>, wk10 <dbl>,
## #   wk11 <dbl>, wk12 <dbl>, wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>,
## #   wk17 <dbl>, wk18 <dbl>, wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>,
## #   wk23 <dbl>, wk24 <dbl>, wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>,
## #   wk29 <dbl>, wk30 <dbl>, wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>,
## #   wk35 <dbl>, wk36 <dbl>, wk37 <dbl>, wk38 <dbl>, wk39 <dbl>, wk40 <dbl>,
## #   wk41 <dbl>, wk42 <dbl>, wk43 <dbl>, wk44 <dbl>, wk45 <dbl>, wk46 <dbl>,
## #   wk47 <dbl>, wk48 <dbl>, wk49 <dbl>, wk50 <dbl>, wk51 <dbl>, wk52 <dbl>,
## #   wk53 <dbl>, wk54 <dbl>, wk55 <dbl>, wk56 <dbl>, wk57 <dbl>, wk58 <dbl>,
## #   wk59 <dbl>, wk60 <dbl>, wk61 <dbl>, wk62 <dbl>, wk63 <dbl>, wk64 <dbl>,
## #   wk65 <dbl>, wk66 <lgl>, wk67 <lgl>, wk68 <lgl>, wk69 <lgl>, wk70 <lgl>,
## #   wk71 <lgl>, wk72 <lgl>, wk73 <lgl>, wk74 <lgl>, wk75 <lgl>, wk76 <lgl>
```

a. Convert all the week columns into a row for each week for each song (where there is an entry). You should wind up with 5,307 rows

```
billboard %>%
 pivot_longer(cols = starts_with("wk"),
    names_to = "week",
    values_to = "ranking",
    values_drop_na = TRUE ,
    names_prefix = "wk"
  )
```

```
## # A tibble: 5,307 x 7
##     year artist  track                   time   date.entered week  ranking
##    <dbl> <chr>   <chr>                   <time> <date>       <chr>   <dbl>
##  1  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   1          87
##  2  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   2          82
##  3  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   3          72
##  4  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   4          77
##  5  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   5          87
##  6  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   6          94
##  7  2000 2 Pac   Baby Don't Cry (Keep... 04:22  2000-02-26   7          99
##  8  2000 2Ge+her The Hardest Part Of ... 03:15  2000-09-02   1          91
##  9  2000 2Ge+her The Hardest Part Of ... 03:15  2000-09-02   2          87
## 10  2000 2Ge+her The Hardest Part Of ... 03:15  2000-09-02   3          92
## # ... with 5,297 more rows
```

b. Figure out the dates corresponding to each week on the chart

```
billboard_revised <- billboard %>%
 pivot_longer(cols = starts_with("wk"), names_to = "week",
              values_to = "ranking", values_drop_na = TRUE ,names_prefix = "wk") %>%
  select(year:time, week, ranking, date.entered ) %>%
  rename(date = date.entered) %>%
  mutate(week = parse_number(week),
         date = date+7*(week-1))

billboard_revised
```

```
## # A tibble: 5,307 x 7
##     year artist  track                   time    week ranking date
##    <dbl> <chr>   <chr>                   <time> <dbl>   <dbl> <date>
##  1  2000 2 Pac   Baby Don't Cry (Keep... 04:22      1      87 2000-02-26
##  2  2000 2 Pac   Baby Don't Cry (Keep... 04:22      2      82 2000-03-04
##  3  2000 2 Pac   Baby Don't Cry (Keep... 04:22      3      72 2000-03-11
##  4  2000 2 Pac   Baby Don't Cry (Keep... 04:22      4      77 2000-03-18
##  5  2000 2 Pac   Baby Don't Cry (Keep... 04:22      5      87 2000-03-25
##  6  2000 2 Pac   Baby Don't Cry (Keep... 04:22      6      94 2000-04-01
##  7  2000 2 Pac   Baby Don't Cry (Keep... 04:22      7      99 2000-04-08
##  8  2000 2Ge+her The Hardest Part Of ... 03:15      1      91 2000-09-02
##  9  2000 2Ge+her The Hardest Part Of ... 03:15      2      87 2000-09-09
## 10  2000 2Ge+her The Hardest Part Of ... 03:15      3      92 2000-09-16
## # ... with 5,297 more rows
```

```
# week 1 = week+7*0
# week 2 = week+7*1
# week 3 = week+7*2

# week+7 *(week-1)
```

c. Sort the data by artist, track and week. Here are what your first entries should be (formatting can be different):

```
billboard_revised %>%
  arrange(artist,track,week)
```

```
## # A tibble: 5,307 x 7
##     year artist  track                   time    week ranking date
##    <dbl> <chr>   <chr>                   <time> <dbl>   <dbl> <date>
##  1  2000 2 Pac   Baby Don't Cry (Keep... 04:22      1      87 2000-02-26
##  2  2000 2 Pac   Baby Don't Cry (Keep... 04:22      2      82 2000-03-04
##  3  2000 2 Pac   Baby Don't Cry (Keep... 04:22      3      72 2000-03-11
##  4  2000 2 Pac   Baby Don't Cry (Keep... 04:22      4      77 2000-03-18
##  5  2000 2 Pac   Baby Don't Cry (Keep... 04:22      5      87 2000-03-25
##  6  2000 2 Pac   Baby Don't Cry (Keep... 04:22      6      94 2000-04-01
##  7  2000 2 Pac   Baby Don't Cry (Keep... 04:22      7      99 2000-04-08
##  8  2000 2Ge+her The Hardest Part Of ... 03:15      1      91 2000-09-02
##  9  2000 2Ge+her The Hardest Part Of ... 03:15      2      87 2000-09-09
## 10  2000 2Ge+her The Hardest Part Of ... 03:15      3      92 2000-09-16
## # ... with 5,297 more rows
```

3. Import and tidy the Iris dataset from http://archive.ics.uci.edu/ml/datasets/Iris. You need two files to generate the data set: iris.data and iris,names. Both are text files. Then plot the measurements using boxplots with the x variable being the species, faceting by plant part (sepal or petal) and by measure dimension (length or width). Your plot should look something like this:

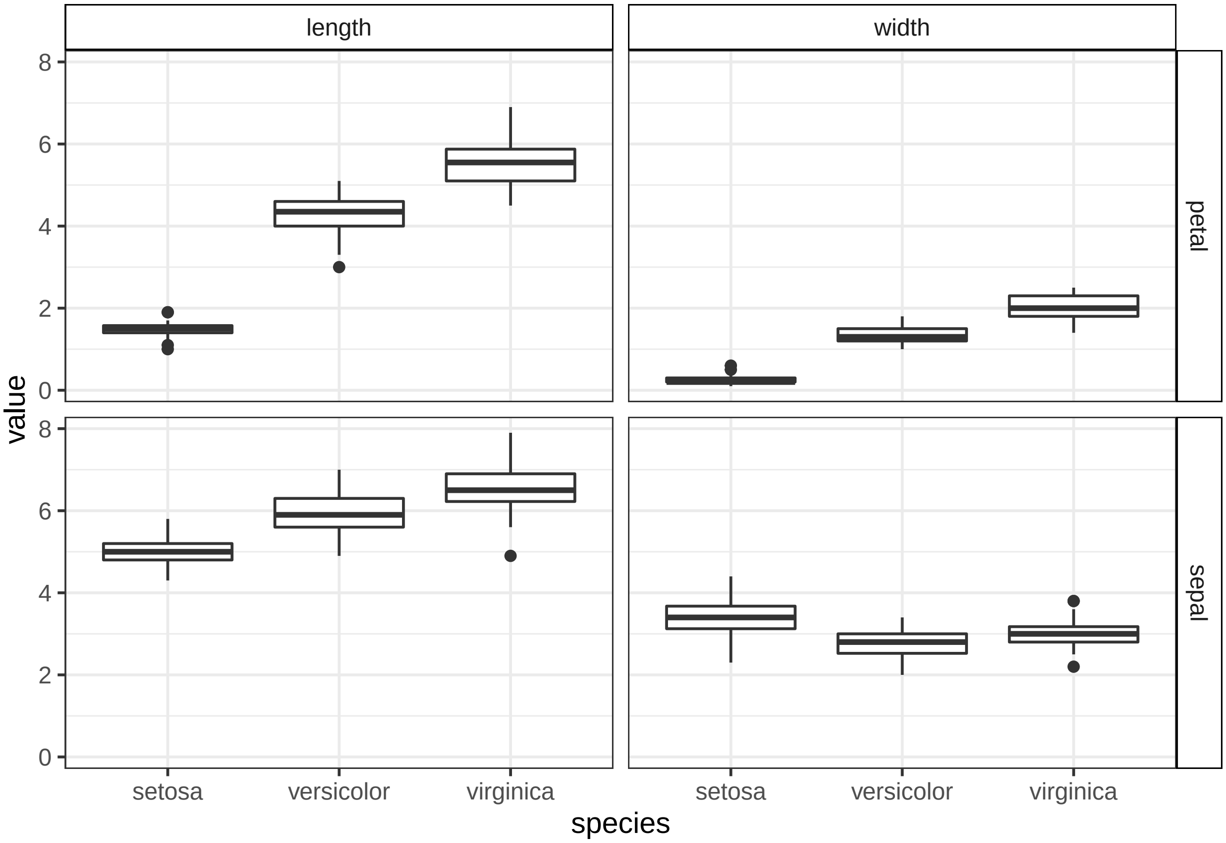
```
iris_data <- read.csv(file = "./data/iris_data.csv", header = FALSE, sep = ",")
iris_real_data <- iris_data %>%

  rename("sepal_length" = "V1", "speal_width" = "V2",
         "petal_length" = "V3", "petal_width" = "V4", "species" = "V5") %>%
  #rename(new = old)

  pivot_longer(cols = sepal_length:petal_width,
               names_to = "cm", values_to = "value") %>%
  separate(col = cm, into = c("sp","dimension")) %>%
  mutate(species = recode(species, "Iris-setosa" = "setosa",
                          "Iris-versicolor" = "versicolor",
                          "Iris-virginica" = "virginica"))


iris_real_data$sp[iris_real_data$sp == "speal"] <- "sepal"

iris_real_data %>%
  ggplot(aes(x = species,y= value)) +
  geom_boxplot()+
  facet_grid(sp~dimension) + # two category uses fact_grid
  theme_bw()+
  theme(strip.background = element_rect(colour = "black", fill = "white"))
```


length
width
petal
sepal
value
species
0
2
4
6
8
0
2
4
6
8
setosa
versicolor
virginica
setosa
versicolor
virginica