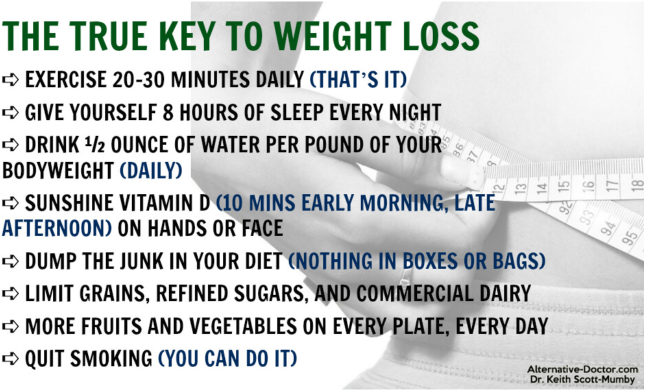
Case Study 7

Sonia Mazzi

03/10/2018

CASE STUDY. A study on the effects of exercise on health with data in four flat files



THE TRUE KEY TO WEIGHT LOSS

- ➪ EXERCISE 20-30 MINUTES DAILY (THAT'S IT)
- ➪ GIVE YOURSELF 8 HOURS OF SLEEP EVERY NIGHT
- ➪ DRINK ½ OUNCE OF WATER PER POUND OF YOUR BODYWEIGHT (DAILY)
- ➪ SUNSHINE VITAMIN D (10 MINS EARLY MORNING, LATE AFTERNOON) ON HANDS OR FACE
- ➪ DUMP THE JUNK IN YOUR DIET (NOTHING IN BOXES OR BAGS)
- ➪ LIMIT GRAINS, REFINED SUGARS, AND COMMERCIAL DAIRY
- ➪ MORE FRUITS AND VEGETABLES ON EVERY PLATE, EVERY DAY
- ➪ QUIT SMOKING (YOU CAN DO IT)

Alternative-Doctor.com
Dr. Keith Scott-Mumby

- Assess the effectiveness of a new exercise programme which is believed to induce weight loss and improve self-rated health.
- 5000 people randomized into two groups: 2,500 people to the treatment and another 2,500 to the control group.
- Health outcomes, weight and self-rated health, were measured before and after the intervention.

The data was provided in 4 files:

- EXER_age_sex_race.csv : Subject demographics at baseline: age, sex, and race (we have seen this data when learning how to import comma separated values file)
- EXER_SRH.csv : Self-Rated Health (SRH) on an ordinal scale.
- EXER_weight_trt : Weight, in pounds, for Treatment Group.
- EXER_weight_con : Weight, in pounds, for Control Group.

Import these files using the `read_csv()` function and explore the variables they contain.

```
data1 = read_csv("DataFiles/EXER_age_sex_race.csv")
```

```
glimpse(data1)
```

```
## Observations: 5,000
## Variables: 2
## $ subject_ID  <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ SexAge_Race <chr> "MALE41.2_White", "FEMALE42.9_White", "FEMALE38.5_...
```

- There are 5000 subjects in the study.
- All the information about each subject is in one column. Separate the information.
- There are missing values for race.

```
data2 = read_csv("DataFiles/EXER_SRH.csv")
```

```
glimpse(data2)
```

```
## Observations: 10,035
## Variables: 4
## $ id   <int> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10,...
## $ trt  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ TIME <chr> "POST", "PRE", "PRE", "POST", "PRE", "POST", "PRE", "POST...
## $ SRH  <chr> "Poor", "Good", "Poor", "Very Poor", "Satisfactory", "Goo...
```

```
unique(as.factor(data2$trt))
```

```
## [1] 1 0
## Levels: 0 1
```

- This file contains the patient id (note the different column name from the previous file),
- and each participant's self rated health before and after the study.
- `trt` takes on two values: 0 (individual didn't exercise, control), 1 (individual exercised, treatment).
- `data2` has 10035 rows but there should only be 10000 rows.

```
data3 = read_csv("DataFiles/EXER_weight_trt.csv")
```

```
glimpse(data3)
```

```
## Observations: 5,017
## Variables: 3
## $ Id          <int> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9,...
## $ PRE_WEIGHT  <dbl> 135.2510, NA, 154.8713, NA, 128.1951, NA, 183.4600...
## $ POST_WEIGHT <dbl> NA, 125.6678, NA, 153.9882, NA, 115.5969, NA, 177....
```

- Information about the weight of patients who received the exercise plan.
- Patient id column has different name than in the previous two data sets,
- Pre- and post-intervention weight.
- Many NAs! if pre-intervention weight is recorded, post-intervention weight is an NA, and vice-versa.
- Expect 5000 observations, but there are 5017.

```
data4 = read_csv("DataFiles/EXER_weight_con.csv")
```

```
glimpse(data4)
```

```
## Observations: 5,012
## Variables: 3
## $ obs_ID      <int> 2501, 2501, 2502, 2502, 2503, 2503, 2504, 2504, 25...
## $ PRE_WEIGHT  <dbl> 159.7587, NA, 176.1611, NA, 181.3907, NA, 175.6615...
## $ POST_WEIGHT <dbl> NA, 158.6920, NA, 174.8270, NA, 179.9042, NA, 175....
```

- Information about the weight of patients who are in the control group.
- Patient id column has different name than in the previous two data sets,
- Pre- and post-intervention weight.
- Many NAs! if pre-intervention weight is recorded, post-intervention weight is an NA, and vice-versa.
- Expect 5000 observations, but there are 5012.

In this study:

- an observational unit is a patient,
- fixed variables are age, sex and race,
- measured variables are self-rated health and weight at two time points, before and after the intervention.

This is messy data:

- a single observational unit is stored in multiple tables,
- multiple variables are stored in one column,
- there is possibly duplicated information,
- and more messy features to be discovered!

- The strategy is to deal with the data sets one by one.
- Manipulate each of them so that they can be joined.
- Have to join by subject id so let us name that column `id` in all four data frames

First data set (data1)

```
glimpse(data1)
```

```
## Observations: 5,000
## Variables: 2
## $ subject_ID  <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ SexAge_Race <chr> "MALE41.2_White", "FEMALE42.9_White", "FEMALE38.5_...
```

```
# data1_1 will be the modified version of data1.
data1_1 = data1
```

```
#add a _ after MALE to separate sex and age
data1_1$SexAge_Race =
  str_replace_all(data1_1$SexAge_Race, "MALE", "MALE_")
```

```
#separate values between _
data1_1 =
separate(data1_1, SexAge_Race, c("Sex","Age","Race"), sep = "_")
```

```r
glimpse(data1_1)
```

```
## Observations: 5,000
## Variables: 4
## $ subject_ID <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...
## $ Sex        <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FE...
## $ Age        <chr> "41.2", "42.9", "38.5", "35.6", "48.5", "36.9", "28...
## $ Race       <chr> "White", "White", "White", "Hispanic", "White", "NA...
```

- Note `Age` is of type character. Should be numeric.

```r
#make Age a numeric variable
data1_1$Age = as.numeric(data1_1$Age)
```

```r
#rename the first column
names(data1_1)[1] = "id"
```

```r
glimpse(data1_1)
```

```
## Observations: 5,000
## Variables: 4
## $ id   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17...
## $ Sex  <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE",...
## $ Age  <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 37....
## $ Race <chr> "White", "White", "White", "Hispanic", "White", "NA", "Wh...
```

- Note that the `Race` column of `data1_1` has the value “NA”, which R reads as a character string and not as a missing value.
- Need to transform the entries “NA” into `NA`.

The function `str_detect` (in package `stringr`) has two arguments

- The 1st argument is a vector where the positions of a string will be detected
- The 2nd argument is the string to be detected

```
#aux contains the position where the string "NA" was found
aux = str_detect(data1_1$Race, "NA")
```

```
#turn those entries into missing values
data1_1$Race[aux] = NA
```

```
glimpse(data1_1)
```

```
## Observations: 5,000
## Variables: 4
## $ id   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17...
## $ Sex  <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE",...
## $ Age  <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 37....
## $ Race <chr> "White", "White", "White", "Hispanic", "White", NA, "Whit...
```

```r
summary(as.factor(data1_1$Race))
```

```
##    Asian    Black Hispanic    White     NA's
##      450      424      846     2876      404
```

```r
summary(as.factor(data1_1$Sex))
```

```
## FEMALE   MALE
##   2556   2444
```

```r
summary(data1_1$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.10   33.40   38.80   38.89   44.10   69.30
```

Second data set (data2)

```
glimpse(data2)
```

```
## Observations: 10,035
## Variables: 4
## $ id   <int> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10,...
## $ trt  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ TIME <chr> "POST", "PRE", "PRE", "POST", "PRE", "POST", "PRE", "POST...
## $ SRH  <chr> "Poor", "Good", "Poor", "Very Poor", "Satisfactory", "Goo...
```

- There are 10,035 rows. Remove exact duplicate rows.

```
data2_1 = data2
```

```
#remove duplicate rows
data2_1 = distinct(data2)
```

```
nrow(data2_1)
```

```
## [1] 10016
```

- After removing duplicate rows now there are 10,016 rows.

- Summaries of columns. Is there anything unusual?

```r
summary(as.factor(data2_1$trt))
```

```
##    0    1 
## 5008 5008
```

```r
summary(as.factor(data2_1$TIME))
```

```
## POST  PRE 
## 5003 5013
```

```r
summary(as.factor(data2_1$SRH))
```

```
##    Excellent         Good         Poor Satisfactory   Very  Poor 
##         2947         1570         1711         1411           16 
##    Very Poor 
##         2361
```

- We spotted something unusual! The SRH level `Very  Poor` appears also with a double space between Very and Poor.
- This happens for 16 rows.

```r
data2_1$SRH =
  str_replace_all(data2_1$SRH, "Very  Poor", "Very Poor")
```

```r
summary(as.factor(data2_1$SRH))
```

```
##     Excellent         Good         Poor Satisfactory    Very Poor
##          2947         1570         1711         1411         2377
```

```r
#remove duplicate rows
data2_1 = distinct(data2_1)
```

```r
glimpse(data2_1)
```

```
## Observations: 10,000
## Variables: 4
## $ id   <int> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10,...
## $ trt  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ TIME <chr> "POST", "PRE", "PRE", "POST", "PRE", "POST", "PRE", "POST...
## $ SRH  <chr> "Poor", "Good", "Poor", "Very Poor", "Satisfactory", "Goo...
```

- There are now two rows per patient.

- Merge all four data sets into one.
- The first data set has one row per observation. The second data set has two rows per observation.
- Spread the `TIME` column, to create two new columns with PRE-SRH and POST-SRH.

```r
data2_1 = spread(data2_1, TIME, SRH)
```

```r
glimpse(data2_1)
```

```
## Observations: 5,000
## Variables: 4
## $ id   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17...
## $ trt  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ POST <chr> "Poor", "Very Poor", "Good", "Good", "Poor", "Excellent",...
## $ PRE  <chr> "Good", "Poor", "Satisfactory", "Poor", "Poor", "Good", "...
```

- Change the names of the 3rd and 4th columns to POST_SRH and PRE_SRH and swap their order.

```r
names(data2_1)[3:4] = c("POST_SRH", "PRE_SRH")
```

```r
data2_1 = data2_1[,c(1,2,4,3)]
```

```r
glimpse(data2_1)
```

```
## Observations: 5,000
## Variables: 4
## $ id       <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
## $ trt      <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ PRE_SRH  <chr> "Good", "Poor", "Satisfactory", "Poor", "Poor", "Good...
## $ POST_SRH <chr> "Poor", "Very Poor", "Good", "Good", "Poor", "Excelle...
```

Third data set (data3)

- Pre- and post-weight of treatment patients.
- Change the name of the first column to id.

```
glimpse(data3)
```

```
## Observations: 5,017
## Variables: 3
## $ Id          <int> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9,...
## $ PRE_WEIGHT  <dbl> 135.2510, NA, 154.8713, NA, 128.1951, NA, 183.4600...
## $ POST_WEIGHT <dbl> NA, 125.6678, NA, 153.9882, NA, 115.5969, NA, 177....
```

```
data3_1 = data3
```

```
names(data3_1)[1] = "id"
```

- There should be 5,000 rows in data3 but there are 5,017 rows.

```r
#remove duplicate rows
data3_1 = distinct(data3_1)
```

```r
nrow(data3_1)
```

```
## [1] 5017
```

- There are no exact duplicate rows in `data3`.
- Let us see if there are more than two records per patient.
- `count()`, in `dplyr`, used to count how many instances for each patient id there are.

Example

```
#count the instances of each patient id in the first 8 rows
aux1 = count(data3_1[1:8,],id)
aux1
```

```
## # A tibble: 4 x 2
##      id     n
##   <int> <int>
## 1     1     2
## 2     2     2
## 3     3     2
## 4     4     2
```

- The result has two columns. One is `id` and the other one is the corresponding count, `n`.

```r
#count the instances of each patient id
aux1 = count(data3_1,id)
```

```r
aux2 = which(aux1$n > 2)
```

```r
aux3 = aux1$id[aux2]
aux3
```

```
## [1] 2394 2395 2396 2397 2398 2399 2400 2401 2402
```

- aux3 gives the patient id's with more than two entries. Let us explore the data for those patients.

```
ff = filter(data3_1, id %in% aux3)
```

```
as.data.frame(ff)
```

```
##      id PRE_WEIGHT POST_WEIGHT
## 1  2394   137.0663          NA
## 2  2394         NA    128.2492
## 3  2394         NA    128.2000
## 4  2395   164.3000          NA
## 5  2395   164.3255          NA
## 6  2395         NA    164.3350
## 7  2395         NA    164.3500
## 8  2396   160.9983          NA
## 9  2396   160.9985          NA
## 10 2396         NA    165.7400
## 11 2396         NA    165.7270
## 12 2397   147.7983          NA
## 13 2397   147.7990          NA
## 14 2397         NA    137.8439
## 15 2397         NA    137.8500
## 16 2398   133.1364          NA
## 17 2398   133.1400          NA
## 18 2398         NA    118.9500
## 19 2398         NA    118.9398
## 20 2399   188.8684          NA
## 21 2399   188.9000          NA
## 22 2399         NA    179.1000
## 23 2399         NA    179.0564
## 24 2400   166.1632          NA
## 25 2400   166.2000          NA
## 26 2400         NA    150.2311
## 27 2400         NA    150.2000
## 28 2401   151.6768          NA
## 29 2401   151.7000          NA
## 30 2401         NA    133.3000
## 31 2401         NA    133.2508
```

Data Science Campus

- Truncate the values of weight (pre- and post-). This will hardly affect the results of the study
- This will create exact row duplicates and then eliminate duplicates.
- The function `mutate_all()` in the `dplyr` package is useful to apply a function to each column.
- In this case the function we want to apply is `round()`.

```r
#this call truncates all the columns
data3_1 = mutate_all(data3_1, trunc)
```

```r
glimpse(data3_1)
```

```
## Observations: 5,017
## Variables: 3
## $ id          <dbl> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9,...
## $ PRE_WEIGHT  <dbl> 135, NA, 154, NA, 128, NA, 183, NA, 166, NA, 120, ...
## $ POST_WEIGHT <dbl> NA, 125, NA, 153, NA, 115, NA, 177, NA, 163, NA, 1...
```

- Now, remove duplicate rows

```r
data3_1 = distinct(data3_1)
```

```r
glimpse(data3_1)
```

```
## Observations: 5,000
## Variables: 3
## $ id          <dbl> 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9,...
## $ PRE_WEIGHT  <dbl> 135, NA, 154, NA, 128, NA, 183, NA, 166, NA, 120, ...
## $ POST_WEIGHT <dbl> NA, 125, NA, 153, NA, 115, NA, 177, NA, 163, NA, 1...
```

- Now we have 5000 rows.

- Need one row per patient with id, pre- and post-weight.
- Strategy: select `id` and pre-weight, eliminate the rows where pre-weight is `NA`.
- Similarly with `id` and post-weight.
- Next merge both data frames into one by patient id.

```r
#select all columns except POST_WEIGHT
temp1 = select(data3_1, -POST_WEIGHT)
```

```r
#filter out the NAs in PRE_WEIGHT
temp1 = filter(temp1, is.na(PRE_WEIGHT) == "FALSE")
```

```r
glimpse(temp1)
```

```
## Observations: 2,500
## Variables: 2
## $ id         <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...
## $ PRE_WEIGHT <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 1...
```

```r
#select all columns except PRE_WEIGHT
temp2 = select(data3_1, -PRE_WEIGHT)
```

```r
#filter out the NAs in POST_WEIGHT
temp2 = filter(temp2, is.na(POST_WEIGHT) == "FALSE")
```

```r
glimpse(temp2)
```

```
## Observations: 2,500
## Variables: 2
## $ id         <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
```

```
#join temp1 and temp2
data3_1 = inner_join(temp1,temp2)
```

```
## Joining, by = "id"
```

```
glimpse(data3_1)
```

```
## Observations: 2,500
## Variables: 3
## $ id         <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
```

Fourth data set (data4)

```
glimpse(data4)
```

```
## Observations: 5,012
## Variables: 3
## $ obs_ID      <int> 2501, 2501, 2502, 2502, 2503, 2503, 2504, 2504, 25...
## $ PRE_WEIGHT  <dbl> 159.7587, NA, 176.1611, NA, 181.3907, NA, 175.6615...
## $ POST_WEIGHT <dbl> NA, 158.6920, NA, 174.8270, NA, 179.9042, NA, 175....
```

```
data4_1 = data4
```

```
names(data4_1)[1] = "id"
```

- There are 5,012 rows in this data set when there should be just 5,000.
- Similar inefficient way of recording information as in previous data set.

- Let us first investigate if there are patient with exact duplicate records.

```
data4_1 = distinct(data4_1)
```

```
glimpse(data4_1)
```

```
## Observations: 5,012
## Variables: 3
## $ id          <int> 2501, 2501, 2502, 2502, 2503, 2503, 2504, 2504, 25...
## $ PRE_WEIGHT  <dbl> 159.7587, NA, 176.1611, NA, 181.3907, NA, 175.6615...
## $ POST_WEIGHT <dbl> NA, 158.6920, NA, 174.8270, NA, 179.9042, NA, 175....
```

- There aren't any exact duplicate rows.

Let us investigate if any patients have duplicate records.

```
#counts instances of each patient id in the whole data set
aux1 = count(data4_1,id)
aux2 = which(aux1$n > 2)
aux3 = aux1$id[aux2]
aux3
```

```
## [1] 4980 4981 4982 4983 4984 4985
```

Patients 4980, 4981, 4982, 4983, 4984 and 4985 have more than two records each.

```r
ff = filter(data4_1, id %in% aux3)
```

```r
as.data.frame(ff)
```

```
##      id PRE_WEIGHT POST_WEIGHT
## 1  4980   151.6736          NA
## 2  4980   151.7000          NA
## 3  4980         NA    150.5249
## 4  4980         NA    150.5300
## 5  4981   171.0954          NA
## 6  4981   171.1000          NA
## 7  4981         NA    168.0745
## 8  4981         NA    168.0800
## 9  4982   154.6500          NA
## 10 4982   154.6518          NA
## 11 4982         NA    153.5068
## 12 4982         NA    153.5100
## 13 4983   141.0200          NA
## 14 4983   141.0217          NA
## 15 4983         NA    140.1000
## 16 4983         NA    140.1410
## 17 4984   137.0652          NA
## 18 4984   137.1000          NA
## 19 4984         NA    134.1708
## 20 4984         NA    134.2000
## 21 4985   195.0055          NA
## 22 4985   195.0100          NA
## 23 4985         NA    193.3600
## 24 4985         NA    193.3532
```

As before, truncate the data and eliminate any duplicate rows.

```r
#this call rounds all the columns to the nearest integer
data4_1 = mutate_all(data4_1,trunc)
```

```r
#eliminate duplicate rows
data4_1 = distinct(data4_1)
```

```r
glimpse(data4_1)
```

```
## Observations: 5,000
## Variables: 3
## $ id          <dbl> 2501, 2501, 2502, 2502, 2503, 2503, 2504, 2504, 25...
## $ PRE_WEIGHT  <dbl> 159, NA, 176, NA, 181, NA, 175, NA, 136, NA, 160, ...
## $ POST_WEIGHT <dbl> NA, 158, NA, 174, NA, 179, NA, 175, NA, 137, NA, 1...
```

- We have 5,000 rows now.

- We have to create a data frame with one row per patient, as before.

```
#select all columns except POST_WEIGHT
temp1 = select(data4_1, -POST_WEIGHT)
```

```
#filter out the NAs in PRE_WEIGHT
temp1 = filter(temp1, is.na(PRE_WEIGHT) == "FALSE")
```

```
glimpse(temp1)
```

```
## Observations: 2,500
## Variables: 2
## $ id         <dbl> 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 250...
## $ PRE_WEIGHT <dbl> 159, 176, 181, 175, 136, 160, 153, 163, 161, 178, 1...
```

```r
#select all columns except PRE_WEIGHT
temp2 = select(data4_1, -PRE_WEIGHT)
```

```r
#filter out the NAs in POST_WEIGHT
temp2 = filter(temp2, is.na(POST_WEIGHT) == "FALSE")
```

```r
glimpse(temp2)
```

```
## Observations: 2,500
## Variables: 2
## $ id          <dbl> 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 25...
## $ POST_WEIGHT <dbl> 158, 174, 179, 175, 137, 159, 155, 164, 160, 177, ...
```

```r
data4_1 = inner_join(temp1,temp2)
```

```
## Joining, by = "id"
```

```r
glimpse(data4_1)
```

```
## Observations: 2,500
## Variables: 3
## $ id          <dbl> 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 25...
## $ PRE_WEIGHT  <dbl> 159, 176, 181, 175, 136, 160, 153, 163, 161, 178, ...
## $ POST_WEIGHT <dbl> 158, 174, 179, 175, 137, 159, 155, 164, 160, 177, ...
```

- Next, join the rows of data3_1 and data4_1 to create data3_2 so that all patients are in one data frame.
- First, add a column in data3_1 called trt with all values equal to 1, and a column in data4_1 called trt as well but with all values equal to zero.

```
data3_1$trt = rep(1, nrow(data3_1))
```

```
data4_1$trt = rep(0, nrow(data4_1))
```

```
glimpse(data3_1)
```

```
## Observations: 2,500
## Variables: 4
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
## $ trt         <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
```

```
glimpse(data4_1)
```

```
## Observations: 2,500
## Variables: 4
## $ id          <dbl> 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 25...
## $ PRE_WEIGHT  <dbl> 159, 176, 181, 175, 136, 160, 153, 163, 161, 178, ...
## $ POST_WEIGHT <dbl> 158, 174, 179, 175, 137, 159, 155, 164, 160, 177, ...
## $ trt         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

```r
data3_2 = bind_rows(data3_1,data4_1)
```

```r
glimpse(data3_2)
```

```
## Observations: 5,000
## Variables: 4
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
## $ trt         <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
```

Data Science
Campus

A unified data set

- Merge data1_1, data2_1 and data3_2.

```
data_exer1 = inner_join(data1_1, data2_1)
```

```
## Joining, by = "id"
```

```
data_exer1 = inner_join(data_exer1, data3_2)
```

```
## Joining, by = c("id", "trt")
```

```
glimpse(data_exer1)
```

```
## Observations: 5,000
## Variables: 9
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Sex         <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "F...
## $ Age         <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47...
## $ Race        <chr> "White", "White", "White", "Hispanic", "White", NA...
## $ trt         <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ PRE_SRH     <chr> "Good", "Poor", "Satisfactory", "Poor", "Poor", "G...
## $ POST_SRH    <chr> "Poor", "Very Poor", "Good", "Good", "Poor", "Exce...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
```

- Further narrow the number of variables introducing a variable `Time`, with values `PRE` and `POST`, and gather SRH and Weight.
- To do that, first create one data frame with all the fixed variables and pre- and post-SRH and another data frame with all the fixed variables and pre- and post-WEIGHT.
- Gather the pre- and post- column into Time and SRH or WEIGHT and then we will join both data sets.

```
temp1 = data_exer1[,1:7]
```

```
temp1 = gather(temp1, "Time", "SRH", 6:7)
```

```
temp1$Time = str_replace_all(temp1$Time, "PRE_SRH", "PRE")
```

```
temp1$Time = str_replace_all(temp1$Time, "POST_SRH", "POST")
```

```
glimpse(temp1)
```

```
## Observations: 10,000
## Variables: 7
## $ id   <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17...
## $ Sex  <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE",...
## $ Age  <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 37....
## $ Race <chr> "White", "White", "White", "Hispanic", "White", NA, "Whit...
## $ trt  <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ Time <chr> "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "...
## $ SRH  <chr> "Good", "Poor", "Satisfactory", "Poor", "Poor", "Good", "...
```

```r
temp2 = data_exer1[,c(1:5,8,9)]
```

```r
temp2 = gather(temp2, "Time", "WEIGHT", 6:7)
```

```r
temp2$Time = str_replace_all(temp2$Time, "PRE_WEIGHT", "PRE")
```

```r
temp2$Time = str_replace_all(temp2$Time, "POST_WEIGHT", "POST")
```

```r
glimpse(temp2)
```

```
## Observations: 10,000
## Variables: 7
## $ id     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ Sex    <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE...
## $ Age    <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 3...
## $ Race   <chr> "White", "White", "White", "Hispanic", "White", NA, "Wh...
## $ trt    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ Time   <chr> "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE",...
## $ WEIGHT <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 149, ...
```

- Merge temp1 and temp2

```
data_exer2 = inner_join(temp1, temp2)
```

```
## Joining, by = c("id", "Sex", "Age", "Race", "trt", "Time")
```

```
glimpse(data_exer2)
```

```
## Observations: 10,000
## Variables: 8
## $ id     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ Sex    <chr> "MALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE", "FEMALE...
## $ Age    <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 3...
## $ Race   <chr> "White", "White", "White", "Hispanic", "White", NA, "Wh...
## $ trt    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ Time   <chr> "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE", "PRE",...
## $ SRH    <chr> "Good", "Poor", "Satisfactory", "Poor", "Poor", "Good",...
## $ WEIGHT <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 149, ...
```

- Both `data_exer1` and `data_exer2` are tidy, and it is a matter of preference and how the data will be used to choose a format to work with.
- We will work with both data frames for the purpose of visualising the data.

- The variable `SRH` is an ordinal categorical variable. Let us tell `R` about the ordinal features of `SRH`.
- Also, `Time` is categorical ordinal. `Sex` and `Race` are simply categorical.
- Treatment, `trt`, is also a factor and we will let R know and re-label the levels from 0 to Control and 1 to Treatment.
- We will do this for both data frames `data_exer1` and `data_exer2`.

```
#transform character columns into factors for data_exer1
data_exer1$PRE_SRH = factor(data_exer1$PRE_SRH, levels = c("Very Poor",
           "Poor", "Satisfactory",  "Good","Excellent"),ordered = TRUE)
```

```
data_exer1$POST_SRH = factor(data_exer1$POST_SRH, levels = c("Very Poor",
            "Poor", "Satisfactory", "Good", "Excellent"),ordered = TRUE)
```

```
data_exer1$Sex = as.factor(data_exer1$Sex)
```

```
data_exer1$Race = as.factor(data_exer1$Race)
```

```
data_exer1$trt = factor(as.character(data_exer1$trt), levels = c("0", "1"),
                        labels = c("Control", "Treatment"))
```

```
glimpse(data_exer1)
```

```
## Observations: 5,000
## Variables: 9
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Sex         <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMA...
## $ Age         <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47...
## $ Race        <fct> White, White, White, Hispanic, White, NA, White, H...
## $ trt         <fct> Treatment, Treatment, Treatment, Treatment, Treatm...
## $ PRE_SRH     <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excell...
## $ POST_SRH    <ord> Poor, Very Poor, Good, Good, Poor, Excellent, Exce...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
```

```r
#transform character columns into factors for data_exer2
data_exer2$SRH = factor(data_exer2$SRH, levels = c("Very Poor", "Poor",
                  "Satisfactory",  "Good", "Excellent"), ordered = TRUE)
```

```r
#Time is also an ordered factor
data_exer2$Time = factor(data_exer2$Time, levels = c("PRE", "POST"),
                          ordered = TRUE)
```

```r
data_exer2$Sex = as.factor(data_exer2$Sex)
```

```r
data_exer2$Race = as.factor(data_exer2$Race)
```

```r
data_exer2$trt = factor(as.character(data_exer2$trt), levels = c("0", "1"),
                         labels = c("Control", "Treatment"))
```

```r
glimpse(data_exer2)
```

```
## Observations: 10,000
## Variables: 8
## $ id     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ Sex    <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, M...
## $ Age    <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, 3...
## $ Race   <fct> White, White, White, Hispanic, White, NA, White, Hispan...
## $ trt    <fct> Treatment, Treatment, Treatment, Treatment, Treatment, ...
## $ Time   <ord> PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, ...
## $ SRH    <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excellent, ...
## $ WEIGHT <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 149, ...
```

- `Age` is continuous in nature.
- However, it is not really expected that at each small change in age we would observe changes in treatment effects.
- Also, in case effects change according to age, a change of, say 5 years, at young age will not possibly see the same effect as a change of 5 years at middle or old age.
- Construct age groups.
- Note that we will use `Age` from `data_exer1` because it contains just one row per observational unit.

```r
summary(data_exer1$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.10   33.40   38.80   38.89   44.10   69.30
```

```r
aux = cut(data_exer1$Age, c(18, 20, 30, 40, 50, 60,70))
```

```r
summary(aux)
```

```
## (18,20] (20,30] (30,40] (40,50] (50,60] (60,70]
##      33     649    2133    1779     374      32
```

- Most participants are 30 - 50 years of age.
- No participants are younger than 18 years or older than 70 years.
- Split age into 18-34yrs, 35-40yrs, 41-45yrs, 46-70yrs, so that we don't have an age category over-represented with number of patients.
- Create a new column in both `data_exer1` and `data_exer2` called `Age_cat` which indicates the age group of the patient.

```r
data_exer1$Age_cat = cut(data_exer1$Age, c(18, 34, 39, 44, 70))
```

```r
data_exer2$Age_cat = cut(data_exer2$Age, c(18, 34, 39, 44, 70))
```

```r
summary(data_exer1$Age_cat)
```

```
## (18,34] (34,39] (39,44] (44,70]
##    1345    1223    1162    1270
```

Let us see how the data looks like now

```
glimpse(data_exer1)
```

```
## Observations: 5,000
## Variables: 10
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Sex         <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMA...
## $ Age         <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47...
## $ Race        <fct> White, White, White, Hispanic, White, NA, White, H...
## $ trt         <fct> Treatment, Treatment, Treatment, Treatment, Treatm...
## $ PRE_SRH     <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excell...
## $ POST_SRH    <ord> Poor, Very Poor, Good, Good, Poor, Excellent, Exce...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
## $ Age_cat     <fct> (39,44], (39,44], (34,39], (34,39], (44,70], (34,3...
```

```r
glimpse(data_exer2)
```

```
## Observations: 10,000
## Variables: 9
## $ id      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,...
## $ Sex     <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, ...
## $ Age     <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, ...
## $ Race    <fct> White, White, White, Hispanic, White, NA, White, Hispa...
## $ trt     <fct> Treatment, Treatment, Treatment, Treatment, Treatment,...
## $ Time    <ord> PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE,...
## $ SRH     <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excellent,...
## $ WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 149,...
## $ Age_cat <fct> (39,44], (39,44], (34,39], (34,39], (44,70], (34,39], ...
```

Let us shuffle the columns so that Age and Age_cat are contiguous.

```
data_exer1 = data_exer1[,c(1:3,10,4:9)]
```

```
data_exer2 = data_exer2[,c(1:3,9,4:8)]
```

```
glimpse(data_exer1)
```

```
## Observations: 5,000
## Variables: 10
## $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Sex         <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMA...
## $ Age         <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47...
## $ Age_cat     <fct> (39,44], (39,44], (34,39], (34,39], (44,70], (34,3...
## $ Race        <fct> White, White, White, Hispanic, White, NA, White, H...
## $ trt         <fct> Treatment, Treatment, Treatment, Treatment, Treatm...
## $ PRE_SRH     <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excell...
## $ POST_SRH    <ord> Poor, Very Poor, Good, Good, Poor, Excellent, Exce...
## $ PRE_WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, ...
## $ POST_WEIGHT <dbl> 125, 153, 115, 177, 163, 105, 151, 143, 119, 159, ...
```

```r
glimpse(data_exer2)
```

```
## Observations: 10,000
## Variables: 9
## $ id      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,...
## $ Sex     <fct> MALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, FEMALE, ...
## $ Age     <dbl> 41.2, 42.9, 38.5, 35.6, 48.5, 36.9, 28.9, 24.7, 47.1, ...
## $ Age_cat <fct> (39,44], (39,44], (34,39], (34,39], (44,70], (34,39], ...
## $ Race    <fct> White, White, White, Hispanic, White, NA, White, Hispa...
## $ trt     <fct> Treatment, Treatment, Treatment, Treatment, Treatment,...
## $ Time    <ord> PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE, PRE,...
## $ SRH     <ord> Good, Poor, Satisfactory, Poor, Poor, Good, Excellent,...
## $ WEIGHT  <dbl> 135, 154, 128, 183, 166, 120, 158, 160, 127, 187, 149,...
```

```r
summary(data_exer2$trt)
```

```
##   Control Treatment
##      5000      5000
```

- Let us see a few ways in which we can tabulate some aspects of the data.
- We can use `table()`, `ftable()` (more than two variables) or `xtabs()`.

```
with(data_exer2, table(SRH,Time))
```

```
##                Time
## SRH              PRE POST
##   Very Poor     1321 1040
##   Poor           959  752
##   Satisfactory   611  800
##   Good           736  834
##   Excellent     1373 1574
```

- Without considering any other factor, there are less patients in the `Poor` and `Very Poor` categories after the intervention.
- there are more patients in the `Satisfactory`, `Good` and `Excellent` categories after the intervention than before the intervention.

Let us see if the trend stays when we consider also the patient's gender.

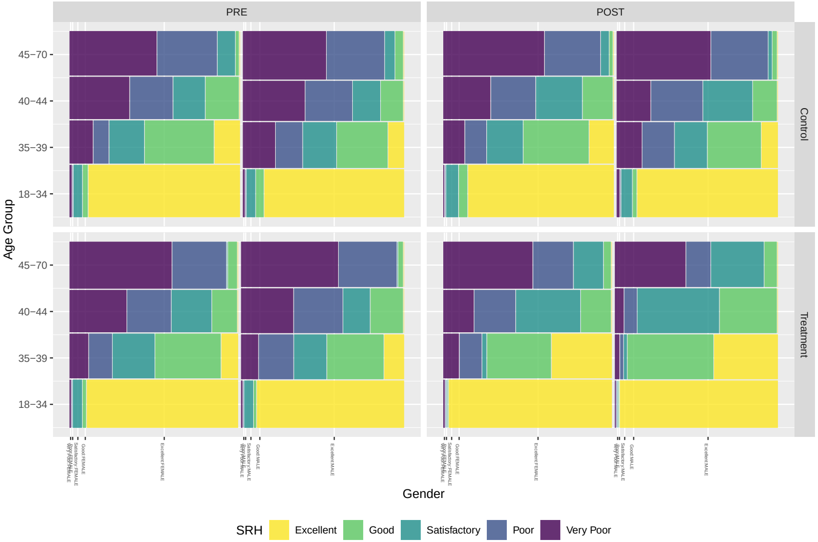
```r
aux2 = with(data_exer2, table(SRH,Time,Sex))
aux2
```

```
## , , Sex = FEMALE
##
##                Time
## SRH              PRE POST
##   Very Poor      660  576
##   Poor           443  414
##   Satisfactory   337  367
##   Good           391  382
##   Excellent      725  817
##
## , , Sex = MALE
##
##                Time
## SRH              PRE POST
##   Very Poor      661  464
##   Poor           516  338
##   Satisfactory   274  433
##   Good           345  452
##   Excellent      648  757
```

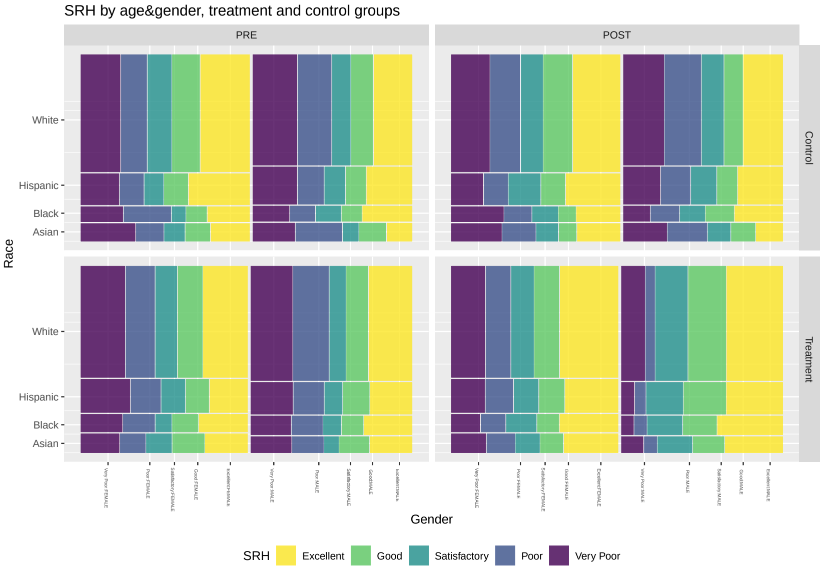
- The trend of positive effect of the exercise plan is still present when we consider gender.
- It seems to be stronger for men than for women.

```
ggplot(data = data_exer2) +
  geom_mosaic(aes( x = product(SRH, Age_cat, Sex), fill=SRH), na.rm=TRUE) +
  theme(axis.text.x=element_text(angle=-90, hjust= .1, size=4)) +
  facet_grid(trt~Time, drop = TRUE) +
  scale_y_continuous("Age Group", breaks = c(0,0.25,0.5, 0.75)+0.25/2 ,
                     labels = c("18-34", "35-39", "40-44", "45-70")) +
  labs(x="Gender",title='SRH by age & gender, treatment and control groups')+
  guides(fill=guide_legend(title = "SRH", reverse = TRUE))+
   theme(legend.position = "bottom")
```

SRH by age & gender, treatment and control groups

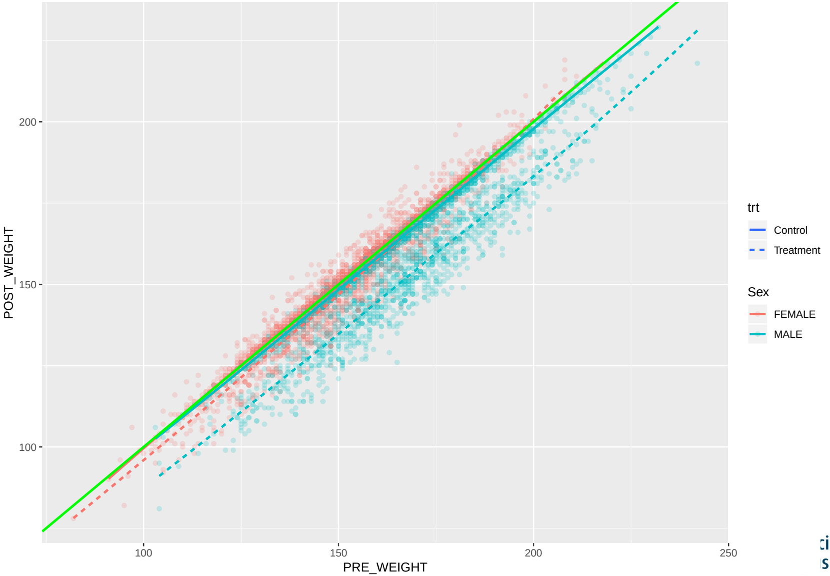


```r
ggplot(data = subset(data_exer2,is.na(Race) == "FALSE")) +
  geom_mosaic(aes( x = product(SRH, Race, Sex), fill=SRH), na.rm=TRUE) +
  theme(axis.text.x=element_text(angle=-90, hjust= .1, size=4)) +
  facet_grid(trt~Time) +
  scale_y_continuous("Race", breaks = c(0.05,0.15,0.3, 0.65) , labels =
                     c("Asian", "Black", "Hispanic", "White")) +
  labs(x="Gender",title='SRH by age&gender, treatment and control groups')+
  guides(fill=guide_legend(title = "SRH", reverse = TRUE))+
  theme(legend.position = "bottom")
```


SRH by age&gender, treatment and control groups
PRE
POST
Control
Treatment
Race
White
Hispanic
Black
Asian
White
Hispanic
Black
Asian
Very Poor:FEMALE
Poor:FEMALE
Satisfactory:FEMALE
Good:FEMALE
Excellent:FEMALE
Very Poor:MALE
Poor:MALE
Satisfactory:MALE
Good:MALE
Excellent:MALE
Very Poor:FEMALE
Poor:FEMALE
Satisfactory:FEMALE
Good:FEMALE
Excellent:FEMALE
Very Poor:MALE
Poor:MALE
Satisfactory:MALE
Good:MALE
Excellent:MALE
Gender
SRH
Excellent
Good
Satisfactory
Poor
Very Poor

- Let us visualise now the measured variables that are continuous, namely Weight.
- Plot `POST_WEIGHT` vs `PRE_WEIGHT` for males (blue) and females (red), before (solid line) and after (dotted line) the intervention.
- Add a 45 degree line through zero (symbolising no treatment effect on weight, i.e. weight before is equal to weight after).

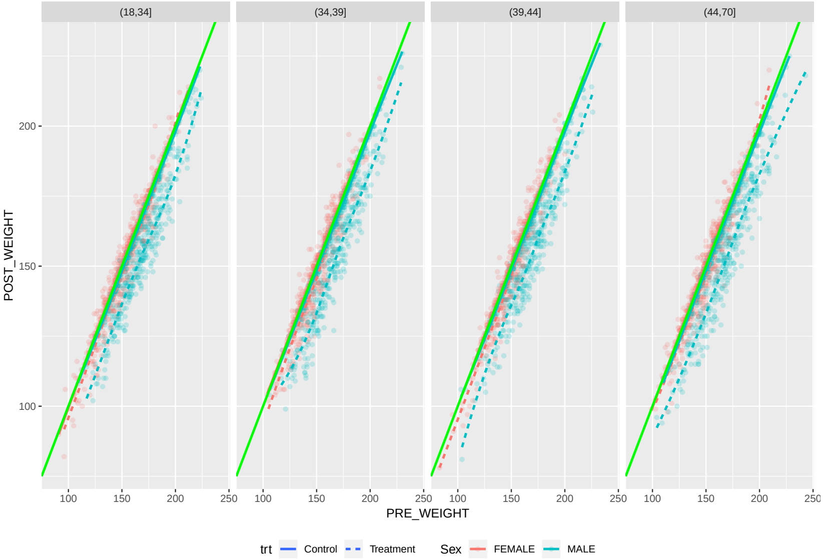
```r
ggplot(data_exer1, aes(x=PRE_WEIGHT, y=POST_WEIGHT, col=Sex, linetype=trt))+
  geom_point(alpha=0.2) +
  stat_smooth(method = "loess", se = FALSE, lwd=1) +
  geom_abline(intercept = 0, slope = 1, color = "green", lwd = 1)
```



- For both sexes the curves of the control patients are nearly overlapping the no-effect green line, as expected.
- The treatment curve for women is very near the no-effect green line. There seems to be a very mild positive effect for women whose weights before the intervention were below 150 pounds.
- The intervention seems to be effective for weight loss for men who underwent the exercise programme, as the blue dotted curve is clearly below the no-effect green line.

- Let us split the data into age categories.

```r
ggplot(data_exer1, aes(x = PRE_WEIGHT, y = POST_WEIGHT, col = Sex,
                       linetype = trt )) +
  geom_point(alpha=0.2) +
  facet_grid(.~Age_cat) +
  stat_smooth(method = "loess", se = FALSE,lwd=1) +
  geom_abline(intercept = 0, slope = 1, color = "green", lwd = 1) +
  theme(legend.position = "bottom")
```


(18,34]
(34,39]
(39,44]
(44,70]
POST_WEIGHT
200
150
100
PRE_WEIGHT
100
150
200
250
trt
Control
Treatment
Sex
FEMALE
MALE

The observations we made before still hold when we split the group by age category. The effect of the exercise programme seems to be more beneficial for men aged 45 or older who weigh over 200 pounds.

Please, complete the short survey

http://www.smartsurvey.co.uk/s/88TEH/