practice_exercise

Ekta Chaudhary

28/06/2020

#Reading the dataset

```
practice_data = read_excel("./data/Practice_exercise.xlsx", sheet = "Data") %>%
  janitor::clean_names() %>%
  select(observation_number,quarter,employee_id, sex = sex_male_1, race, age, hospital_visit = hospital_
  mutate(
    age_cat = case_when(
      age < 30 ~ 1,
      age <= 45 ~ 2,
      age > 45 ~ 3
    )
  )
```

#Checking for missing data

```
sapply(practice_data, function(x) sum(is.na(x)))
```

```
## observation_number             quarter         employee_id
##                  0                   0                   0
##                sex                race                 age
##                 71                2123                   0
##      hospital_visit              salary        health_score
##                  0                   0                   0
##            age_cat
##                  0
```

```
practice_data %>%
  select(everything()) %>%
  summarise_all(funs(sum(is.na(.))))
```

```
## # A tibble: 1 x 10
##   observation_num~ quarter employee_id   sex  race   age hospital_visit
##              <int>   <int>       <int> <int> <int> <int>          <int>
## 1                0       0           0    71  2123     0              0
## # ... with 3 more variables: salary <int>, health_score <int>,
## #   age_cat <int>
```

#Finding the minimum and maximum values of each variable

```
sapply(practice_data, function(x) min(x))
```

```
## observation_number             quarter         employee_id
##       1.000000e+00        1.000000e+00        1.000000e+00
##                sex                race                 age
```

```
##                NA                   NA          7.000000e+00
##     hospital_visit               salary          health_score
##       0.000000e+00         2.835070e+04          6.265991e-01
##           age_cat
##       1.000000e+00
```

```r
sapply(practice_data, function(x) max(x))
```

```
## observation_number              quarter           employee_id
##           19103.00                12.00               2000.00
##                sex                 race                   age
##                 NA                   NA                172.00
##     hospital_visit               salary          health_score
##               1.00             68826.34                 10.00
##            age_cat
##               3.00
```

#Checking the number of employees with health score outside the range of data

```r
practice_data %>%
  count(
   health_sc_6 = ifelse(health_score > 6, 1, 0)
  )
```

```
## # A tibble: 2 x 2
##   health_sc_6     n
##         <dbl> <int>
## 1           0 17865
## 2           1  1238
```

#Calculating the number of quarters for which the employees have missing data on sex

```r
practice_data %>%
  select(
    employee_id, sex
  ) %>%
  filter(
  is.na(sex)
  ) %>%
  group_by(
    employee_id
  ) %>%
  summarise(
    missing = sum(is.na(sex))
  )
```

```
## # A tibble: 7 x 2
##   employee_id missing
##         <dbl>   <int>
## 1        1994      10
## 2        1995       9
## 3        1996      12
```

2

```
## 4          1997        11
## 5          1998        12
## 6          1999         7
## 7          2000        10
```

#Calculating the number of quarters for which the employees have missing data on race

```r
practice_data %>%
  select(
    employee_id, race
  ) %>%
  filter(
    is.na(race)
  ) %>%
  group_by(
    employee_id
  ) %>%
  summarise(
    miss = sum(is.na(race))
  )
```

```
## # A tibble: 220 x 2
##     employee_id  miss
##            <dbl> <int>
##  1            8    10
##  2           10    12
##  3           13     9
##  4           22     9
##  5           36    12
##  6           38    12
##  7           48    10
##  8           49     7
##  9           51     8
## 10           55     9
## # ... with 210 more rows
```

#Calculating the number of employees in each age group for each quarter

```r
emp_data = practice_data %>%
  mutate(
    quarter = factor(
    quarter),
    age_cat = factor(age_cat)
  )
```

```r
emp_data = emp_data %>%
  select(
    employee_id, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  tally()
```

3

```
e <- ggplot(emp_data, aes(x = quarter, y = n, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Number of employees", title = "Number of
e
```

Number of employees in different age group for each quarter

age_cat ─●─ <30 years ─●─ 31–45 years ─●─ >45 years
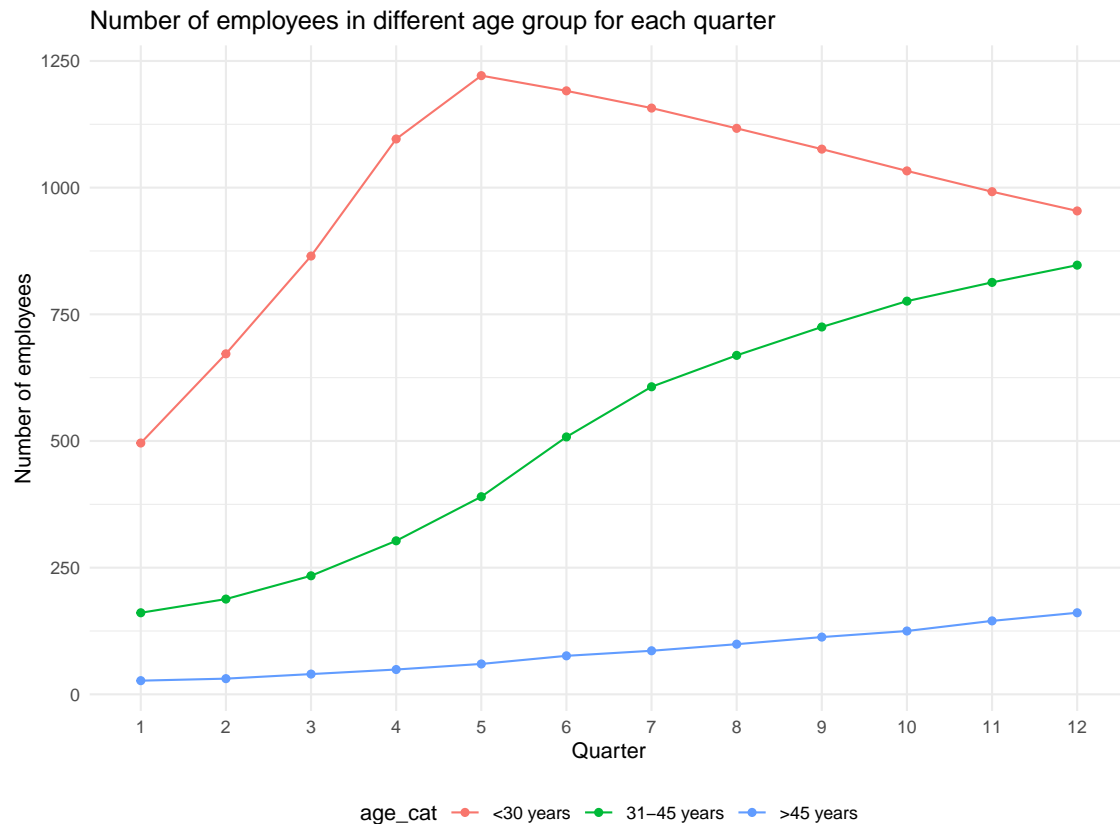
#Checking the trend in average salary over time

```
practice_data %>%
  select(
    salary, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_salary = mean(salary)
  )
```

```
## # A tibble: 12 x 2
##    quarter avg_salary
##      <dbl>      <dbl>
## 1        1     43628.
## 2        2     44274.
## 3        3     45021.
## 4        4     45531.
## 5        5     46133.
## 6        6     46948.
```

```
##  7          7      47780.
##  8          8      48667.
##  9          9      49562.
## 10         10      50498.
## 11         11      51433.
## 12         12      52376.
```

```r
salary_data = practice_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

#Checking the trend in average salary over time by age group

```r
salary_data = salary_data %>%
  select(
    salary, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_salary = mean(salary)
  ) #%>%
  #pivot_wider(
    #names_from = age_cat, values_from = avg_salary
  #)
```

```r
p <- ggplot(salary_data, aes(x = quarter, y = avg_salary, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Average Salary", title = "Trend in average
p
```

Trend in average Salary over time by age group

#Checking the trend in mean health score over time

```
hc = practice_data %>%
  select(
    health_score, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

#Mean health score over time by age group

```
practice_data %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
## # A tibble: 36 x 3
```

```
## # Groups:   quarter [12]
##    quarter age_cat avg_score
##      <dbl>   <dbl>     <dbl>
## 1        1       1      3.21
## 2        1       2      3.90
## 3        1       3      4.56
## 4        2       1      3.38
## 5        2       2      3.62
## 6        2       3      4.95
## 7        3       1      3.41
## 8        3       2      3.63
## 9        3       3      4.16
## 10       4       1      3.34
## # ... with 26 more rows
```

```r
health_sc = practice_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

```r
health_sc = health_sc %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

In group by and summarise, the number of rows is the number of groups, and the columns are the values you have summarised.

```r
p1 <- ggplot(health_sc, aes(x = quarter, y = avg_score, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mean
p1
```

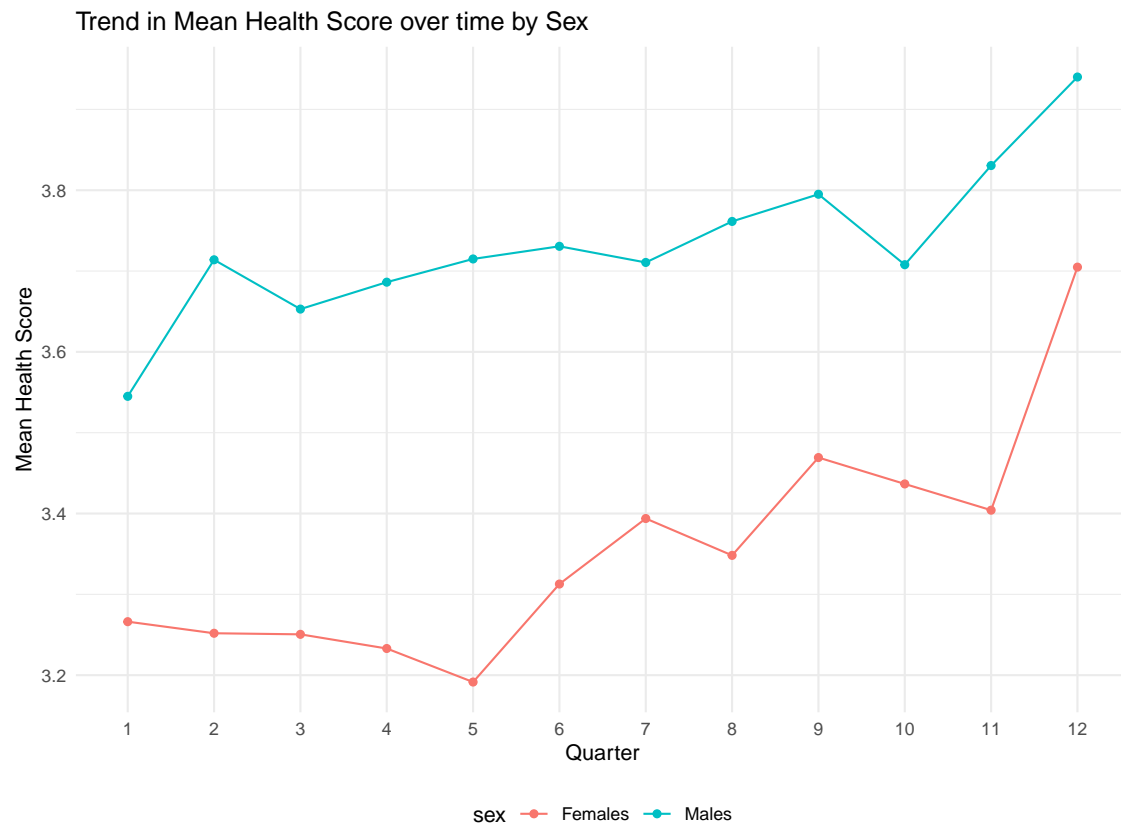Trend in Mean Health Score over time by age group
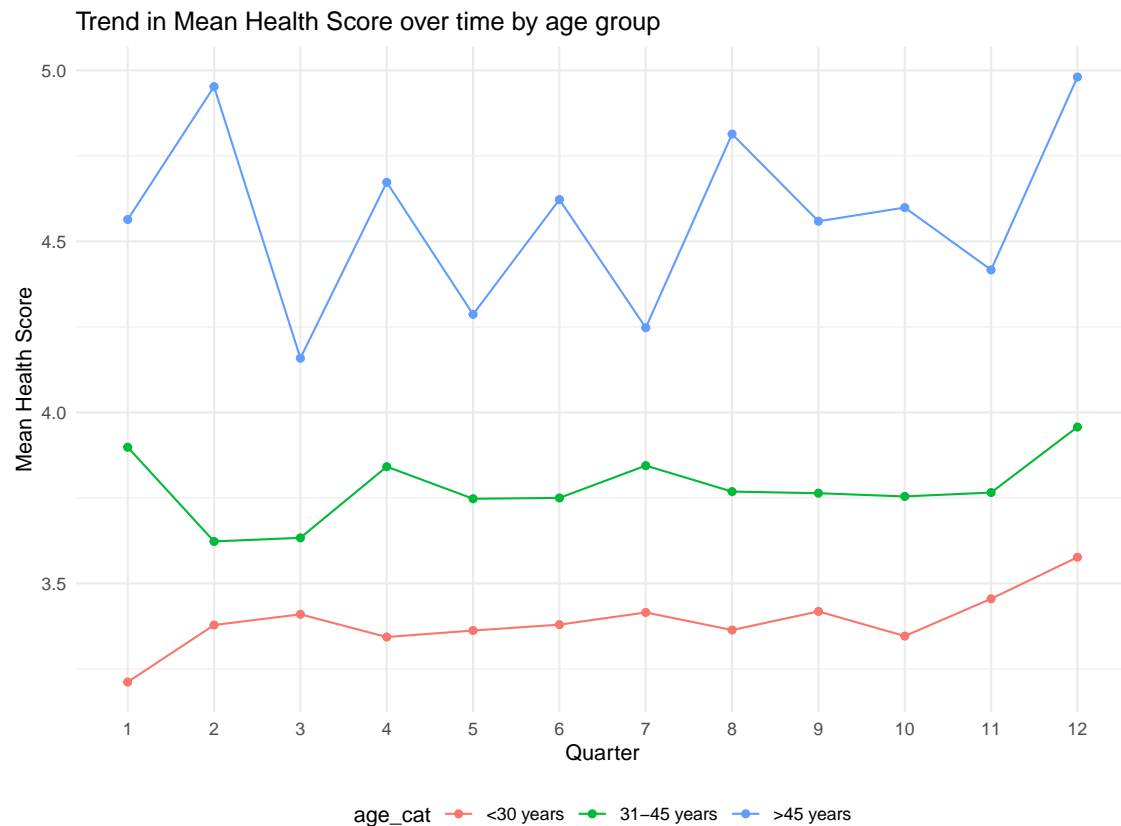
#Mean health score over time by sex

```
health_sex = practice_data %>%
  mutate(
    sex = factor(sex),
quarter = factor(quarter))
```

```
health_sex = health_sex %>%
  drop_na() %>%
  select(
    health_score, quarter, sex
  ) %>%
  group_by(
    quarter, sex
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
pq <- ggplot(health_sex, aes(x = quarter, y = avg_score, group = sex)) +
  geom_line(aes(color = sex)) +
  geom_point(aes(color = sex)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mean Hea
pq
```

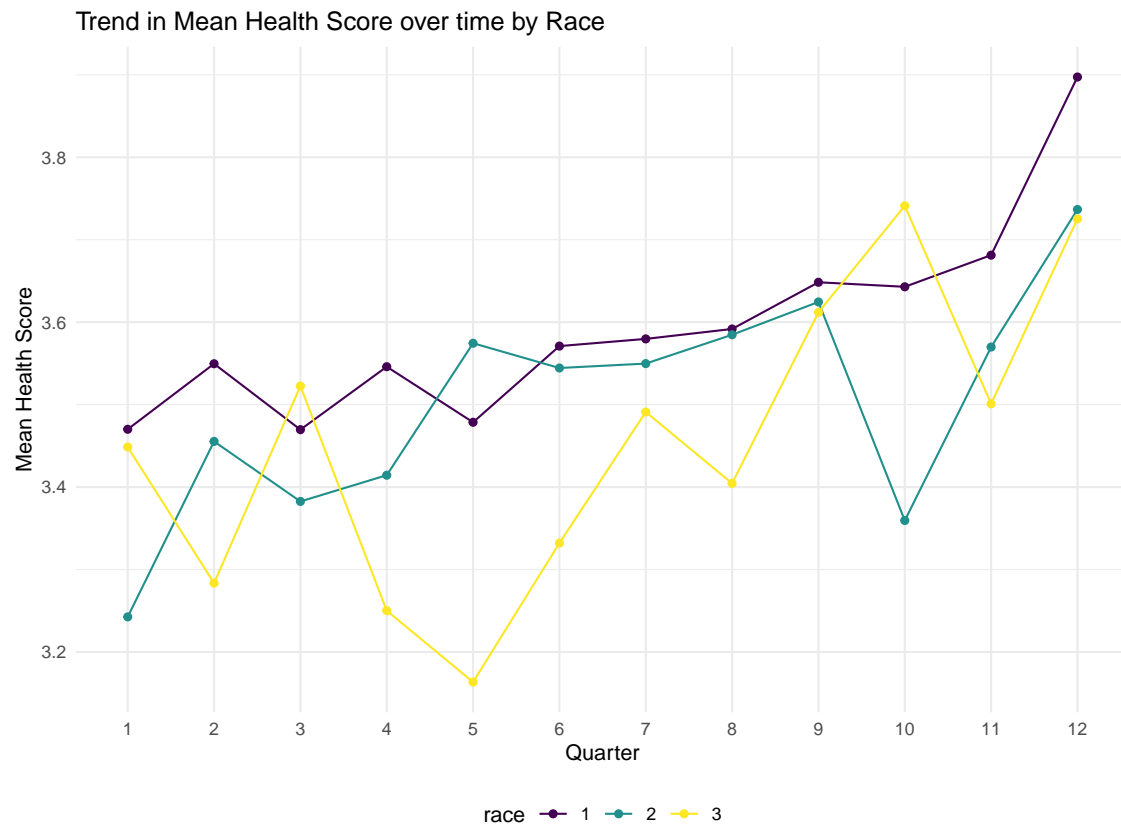Trend in Mean Health Score over time by Sex

#Mean health score over time by race

```
health_race = practice_data %>%
  mutate(
    race = factor(race),
quarter = factor(quarter))
```

```
health_race = health_race %>%
  drop_na() %>%
  select(
    health_score, quarter, race
  ) %>%
  group_by(
    quarter, race
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
pe <- ggplot(health_race, aes(x = quarter, y = avg_score, group = race)) +
  geom_line(aes(color = race)) +
  geom_point(aes(color = race)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mean He
pe
```

9

Trend in Mean Health Score over time by Race

#Correcting the data quality issues

```r
new_data = practice_data %>%
  drop_na() %>%
  filter(
    health_score <= 6,
    age >= 14, age <= 75
  )
```

#Checking the trend in mean health score over time in the corrected data

```r
new_data %>%
  select(
    health_score, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_score = mean(health_score)
  ) %>%
  knitr::kable()
```

quarter	avg_score
1	2.957220
2	3.062881

quarter	avg_score
3	3.053617
4	3.082009
5	3.080370
6	3.121949
7	3.192630
8	3.135304
9	3.177808
10	3.147629
11	3.234287
12	3.298995

#Mean health score over time by age group

```
new_data %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  ) %>%
knitr::kable()
```

quarter	age_cat	avg_score
1	1	2.796936
1	2	3.388951
1	3	3.425028
2	1	2.989041
2	2	3.195572
2	3	3.988790
3	1	2.957211
3	2	3.287013
3	3	3.764033
4	1	2.980156
4	2	3.332257
4	3	3.922990
5	1	2.953362
5	2	3.348889
5	3	3.963946
6	1	2.997311
6	2	3.310626
6	3	3.863035
7	1	3.012671
7	2	3.440885
7	3	3.899116
8	1	2.970070
8	2	3.298552
8	3	4.017854

quarter	age_cat	avg_score
9	1	2.956768
9	2	3.397101
9	3	3.911494
10	1	2.961097
10	2	3.319968
10	3	3.710119
11	1	3.036243
11	2	3.357914
11	3	3.924474
12	1	3.113301
12	2	3.404411
12	3	3.954721

```r
health_score = new_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

```r
health_score = health_score %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```r
pn <- ggplot(health_score, aes(x = quarter, y = avg_score, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mea
pn
```

Trend in Mean Health Score over time by age group
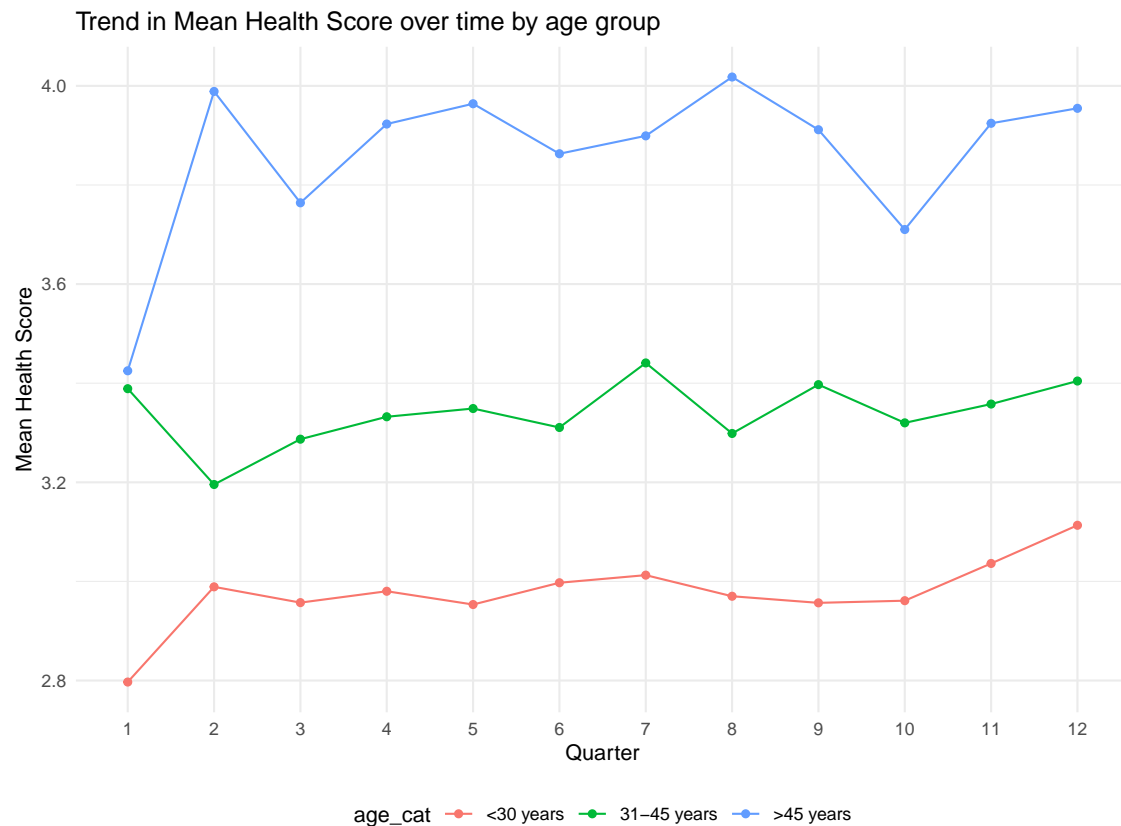
#Calculating the trend in hospital visits over time

```r
hosp = practice_data %>%
  select(
    employee_id, hospital_visit, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    percent = (sum(hospital_visit)/n())*100
  )
```

```r
s <- ggplot(hosp, aes(x = quarter, y = percent, color = percent)) + theme_classic() + geom_line() + geo
  scale_x_discrete(name = "Quarter", limits = c("1","2","3","4","5","6","7","8","9","10","11","12")) +
  scale_y_continuous(name = "Percentage of employees visiting hospitals",
                     breaks = seq(0, 25, 5),
                     limits = c(0, 25)) + labs(x = "Quarter", y = "Percentage of employees visiti
s
```

Trend in Hospital visits over time
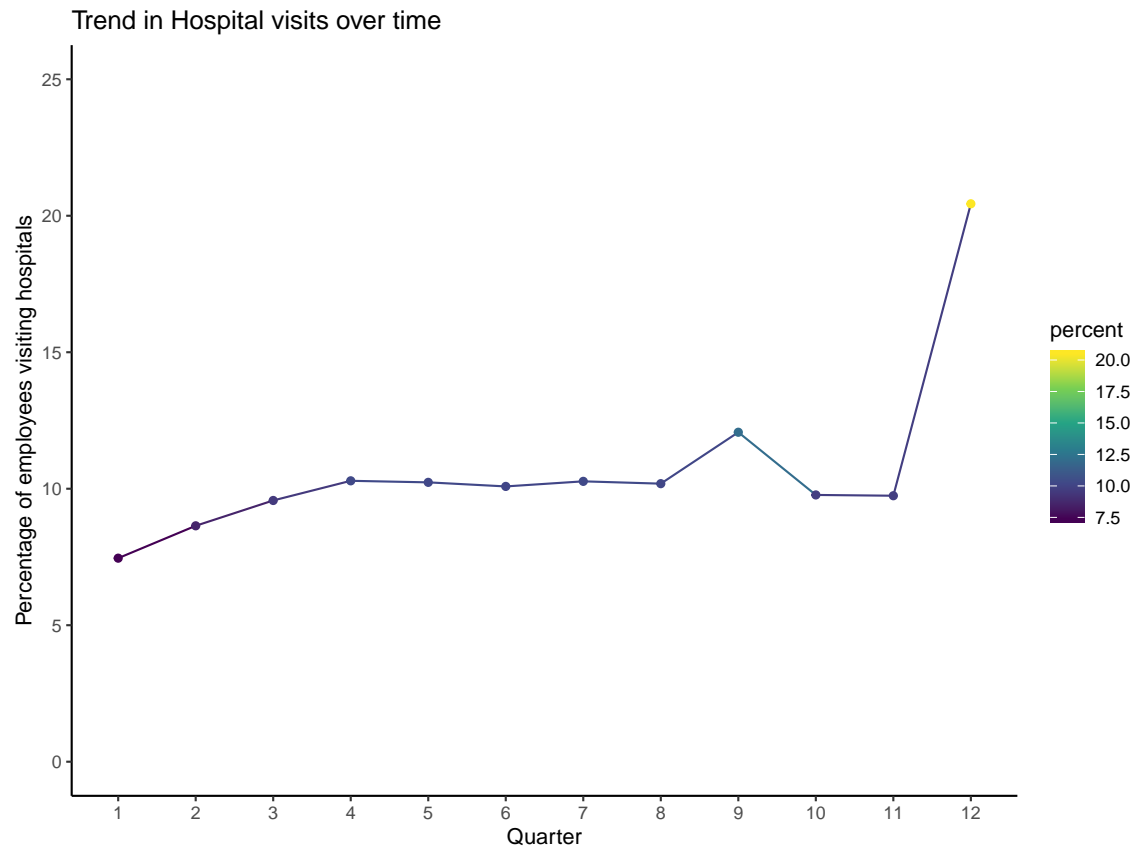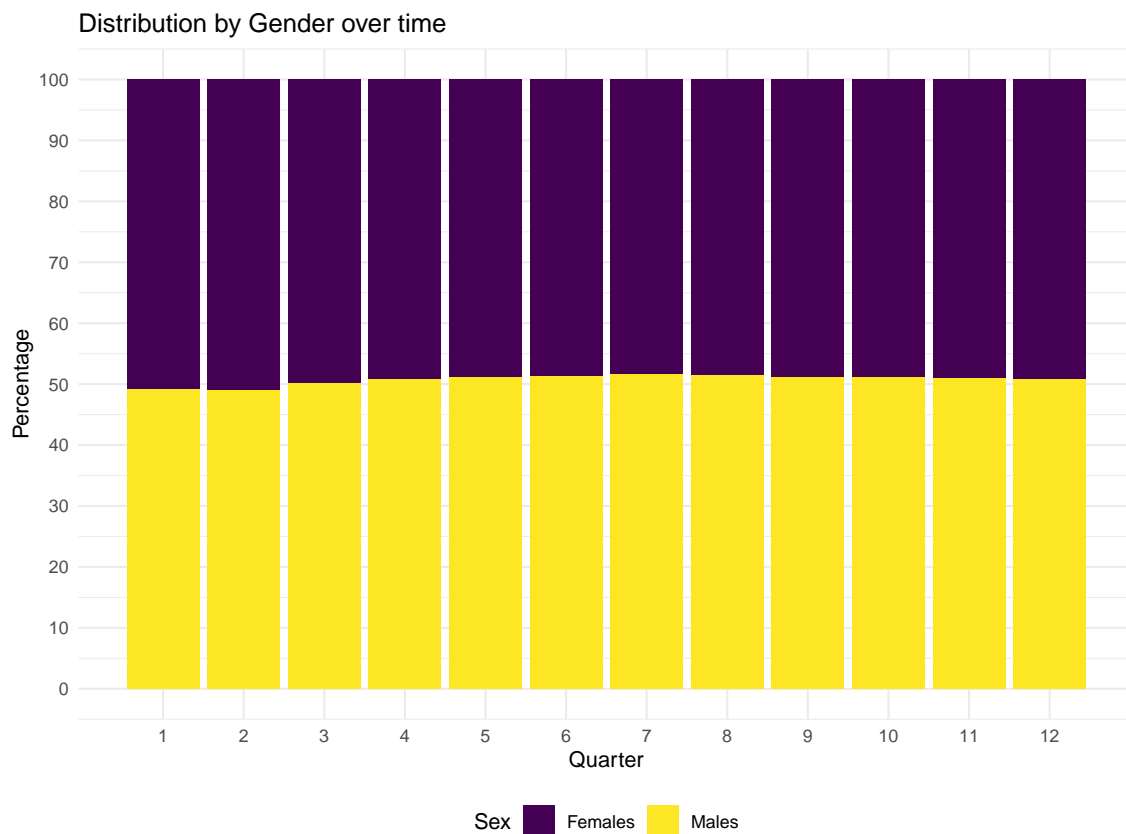
#Calculating the distribution by gender over time

```r
w = practice_data %>%
  drop_na() %>%
  mutate(
    sex = factor(sex)
  ) %>%
  select(
     employee_id, quarter, sex
  ) %>%
  group_by(
    quarter, sex
  ) %>%
summarise(n = n()) %>%
mutate(freq = n / sum(n)*100)
w
```

```
## # A tibble: 24 x 4
## # Groups:   quarter [12]
##    quarter sex        n  freq
##      <dbl> <fct> <int> <dbl>
## 1       1 0      305  50.8
## 2       1 1      295  49.2
## 3       2 0      399  50.9
## 4       2 1      385  49.1
## 5       3 0      503  49.8
## 6       3 1      507  50.2
```

14

```
## 7          4 0          632   49.1
## 8          4 1          655   50.9
## 9          5 0          726   48.9
## 10          5 1          759   51.1
## # ... with 14 more rows
```

```
t = w %>%
  ggplot(aes(x = quarter, y = freq, fill = sex)) + geom_bar(stat = "identity", legend = c("Female", "Mal
  scale_y_continuous(name = "Percentage",
                        breaks = seq(0, 100, 10),
                        limits = c(0, 100)) + labs(x = "Quarter", y = "Percentage" , title = "Distri
t
```


Distribution by Gender over time

#Calculating the distribution by race over time

```
j = practice_data %>%
  drop_na() %>%
  mutate(
    race = factor(race)
  ) %>%
  select(
     quarter, race
  ) %>%
  group_by(
    quarter, race
  ) %>%
  summarise(n = n()) %>%
```

```
mutate(freq = n / sum(n)*100)
j
```

```
## # A tibble: 36 x 4
## # Groups:   quarter [12]
##    quarter race      n  freq
##      <dbl> <fct> <int> <dbl>
## 1        1 1       336   56.
## 2        1 2       167  27.8
## 3        1 3        97  16.2
## 4        2 1       432  55.1
## 5        2 2       222  28.3
## 6        2 3       130  16.6
## 7        3 1       561  55.5
## 8        3 2       293  29.0
## 9        3 3       156  15.4
## 10       4 1       718  55.8
## # ... with 26 more rows
```

```
k = j %>%
  ggplot(aes(x = quarter, y = freq, fill = race)) + geom_bar(stat = "identity") +
  theme_classic() + scale_fill_manual(values = c("dodgerblue4","skyblue","pink")) + scale_x_discrete(nam
  scale_y_continuous(name = "Percentage",
                     breaks = seq(0, 100, 10),
                     limits = c(0, 100)) + labs(x = "Quarter", y = "Percentage" , title = "Distrib
k
```


Distribution by Race over time
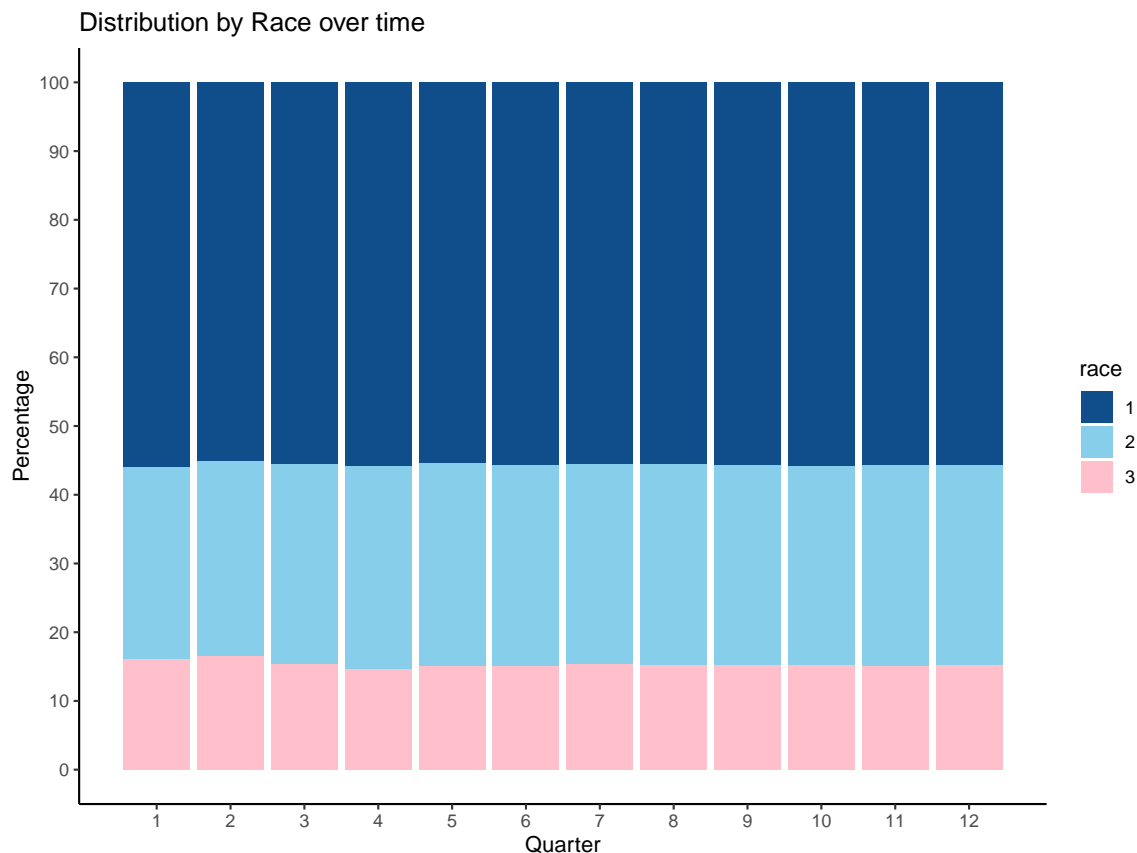
#Scatterplot between salary and health score

```
practice_data %>%
  filter(
    health_score <= 6
  ) %>%
  ggplot(
    aes(x = health_score, y = salary)
  ) + geom_point(aes(color = "yellow")) + labs(
    x = "Heath Score", y = "Salary", title = "Association between Salary and Health Score"
  ) + scale_x_discrete(name = "Quarter", limits = c("1","2","3","4","5","6","7","8","9","10","11","12"))
```


Association between Salary and Health Score

#Box-plot showing the Mean health score for both genders

```
practice_data %>%
  drop_na() %>%
  filter(
    health_score <= 6
  ) %>%
  mutate(
    sex = factor(sex)
  ) %>%
  ggplot(
    aes(
      x = sex, y = health_score
    )
  ) + geom_boxplot()
```

17
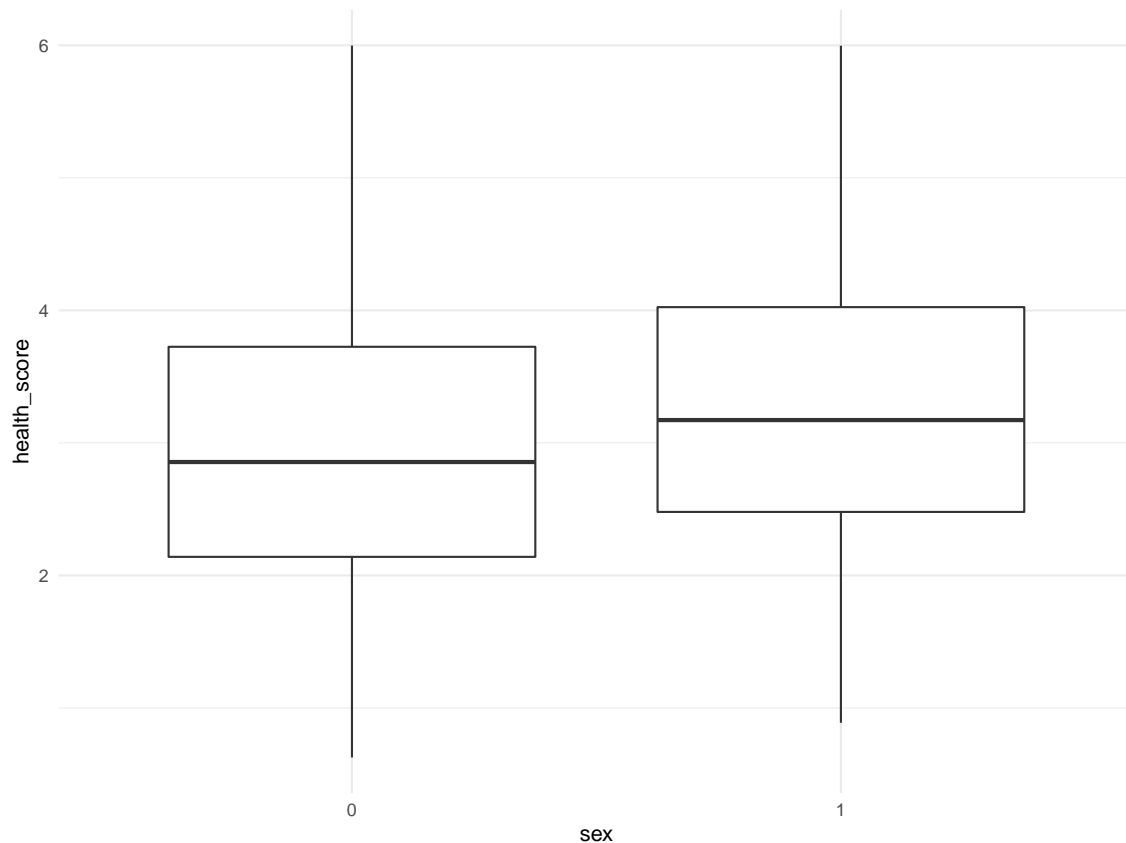
#Checking the association between sex and health score using two sample t-test

```
t.test(health_score ~ sex, data = new_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  health_score by sex
## t = -17.726, df = 15816, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.3343974 -0.2678069
## sample estimates:
## mean in group 0 mean in group 1
##        2.995977        3.297079
```

#Checking the association between hospital visit and health score using two sample t-test

```
t.test(health_score ~ hospital_visit, data = new_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  health_score by hospital_visit
## t = -26.924, df = 2278.4, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
##  -0.7129211 -0.6161208
## sample estimates:
## mean in group 0 mean in group 1
##        3.076413        3.740934
```

#Scatterplot showing the association between age and health score

```
new_data %>%
  filter(
    age <= 60
  ) %>%
ggplot(
  aes(
    x = age, y = health_score
  )
) + geom_point(
  aes(color = "pink")
)
```
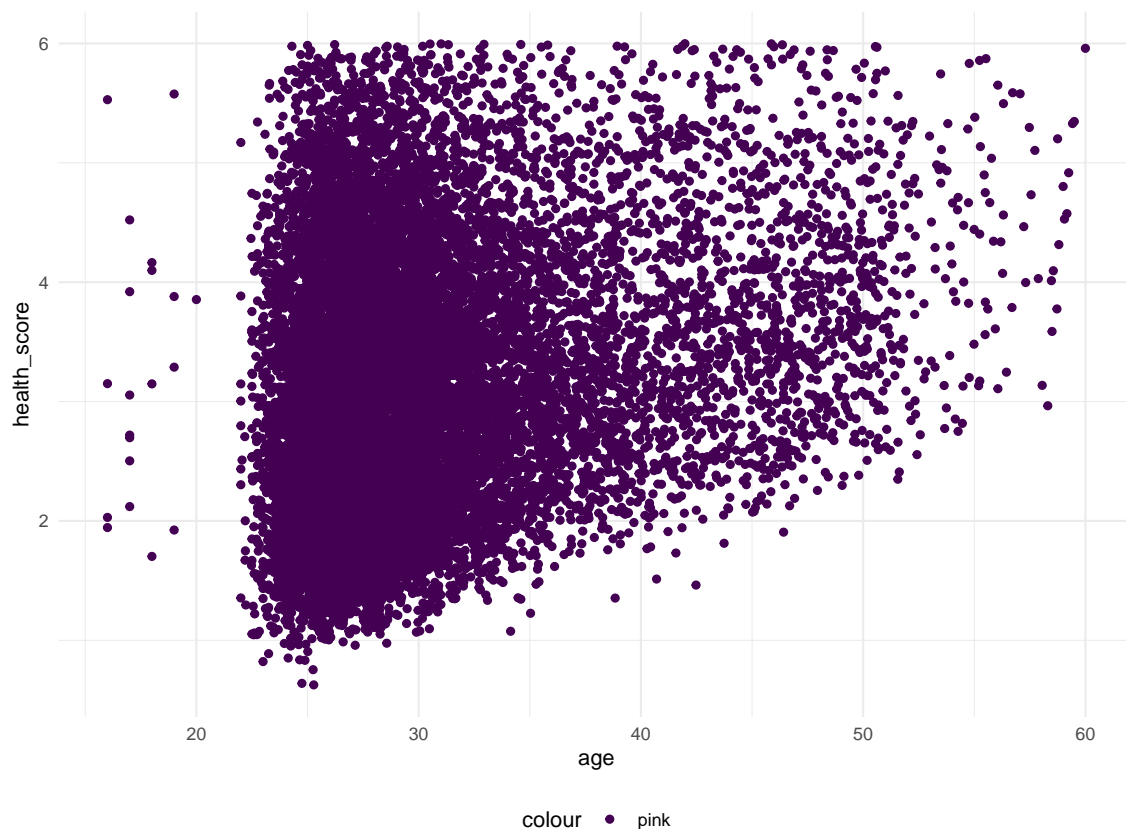

#Checking the association between race and health score

```
av <- aov(health_score ~ race, data = new_data)
summary(av)
```

```
##               Df Sum Sq Mean Sq F value  Pr(>F)
## race           1     12  11.726   10.06 0.00152 **
```

```
## Residuals    15876   18513     1.166
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#Creating 2*2 tables

```r
practice_data %>%
  filter(
    age_cat != 3
  ) %>%
  mutate(
    sal_cat = case_when(
      salary < 40000 ~ "First",
      salary > 40000 ~ "Second"
    )
  ) %>%
  janitor::tabyl(age_cat, sal_cat)
```

```
##  age_cat First Second
##        1   678  11192
##        2   337   5884
```

```r
practice_data %>%
  filter(
    age_cat != 3
  ) %>%
  mutate(
    sal_cat = case_when(
      salary < 40000 ~ "First",
      salary > 40000 ~ "Second"
    )
  ) %>%
  group_by(age_cat,sal_cat) %>%
  summarise(
    n = n()
  ) %>%
  pivot_wider(
    names_from = sal_cat, values_from = n
  )
```

```
## # A tibble: 2 x 3
## # Groups:   age_cat [2]
##   age_cat First Second
##     <dbl> <int>  <int>
## 1       1   678  11192
## 2       2   337   5884
```

#General summaries

```r
practice_data %>%
  group_by(
    quarter
```

```
) %>%
summarise(
  n = n(),
  mean_age = mean(age),
  median_age = median(age),
  mean_salary = mean(salary),
  median_salary = median(salary)
)
```

```
## # A tibble: 12 x 6
##     quarter       n mean_age median_age mean_salary median_salary
##       <dbl> <int>    <dbl>      <dbl>       <dbl>         <dbl>
## 1        1     684     28.8       26.6       43628.        43612.
## 2        2     891     28.6       26.5       44274.        44254.
## 3        3    1139     28.6       26.8       45021.        45002.
## 4        4    1448     28.8       27.1       45531.        45464.
## 5        5    1671     29.3       27.6       46133.        46119.
## 6        6    1775     29.9       28.1       46948.        46915.
## 7        7    1850     30.5       28.5       47780.        47749.
## 8        8    1885     30.9       28.9       48667.        48664.
## 9        9    1914     31.4       29.3       49562.        49603.
## 10      10    1934     31.8       29.6       50498.        50548.
## 11      11    1950     32.2       29.9       51433.        51512.
## 12      12    1962     32.6       30.2       52376.        52440.
```

Notes: Grouped mutate Summarizing collapses groups into single data points. In contrast, using mutate() in conjuntion with group_by() will retain all original data points and add new variables computed within groups.

```
practice_data %>%
  group_by(
    quarter
  ) %>%
  mutate(
    hs = round(health_score, digits = 0)
  )
```

```
## # A tibble: 19,103 x 11
## # Groups:   quarter [12]
##     observation_num~ quarter employee_id   sex  race   age hospital_visit
##                <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
## 1                 1       1           1     1     0     3  27.3              0
## 2                 2       2           2     1     0     3  27.8              0
## 3                 3       3           3     1     0     3  28.1              0
## 4                 4       4           4     1     0     3  28.3              0
## 5                 5       5           5     1     0     3  28.6              0
## 6                 6       6           6     1     0     3  28.8              0
## 7                 7       7           7     1     0     3  29.1              0
## 8                 8       8           8     1     0     3  29.3              0
## 9                 9       9           9     1     0     3  29.6              0
## 10               10      10          10     1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, hs <dbl>
```

Notes: Window functions The previous example used mean() to compute the mean within each group, which was then subtracted from the observed max tempurature. mean() takes n inputs and produces a single output.

Window functions, in contrast, take n inputs and return n outputs, and the outputs depend on all the inputs. There are several categories of window functions; you're most likely to need ranking functions and offsets, which we illustrate below.

```
practice_data %>%
  group_by(quarter, sex) %>%
  mutate(
    salary_rank = min_rank(salary)
  )
```

```
## # A tibble: 19,103 x 11
## # Groups:   quarter, sex [36]
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##               <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1               1       1           1     0     3  27.3              0
##  2               2       2           1     0     3  27.8              0
##  3               3       3           1     0     3  28.1              0
##  4               4       4           1     0     3  28.3              0
##  5               5       5           1     0     3  28.6              0
##  6               6       6           1     0     3  28.8              0
##  7               7       7           1     0     3  29.1              0
##  8               8       8           1     0     3  29.3              0
##  9               9       9           1     0     3  29.6              0
## 10              10      10           1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, salary_rank <int>
```

This sort of ranking is useful when filtering data based on rank

```
practice_data %>%
  group_by(quarter, sex) %>%
    filter(
  min_rank(salary) < 2
    )
```

```
## # A tibble: 36 x 10
## # Groups:   quarter, sex [36]
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##               <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1             3974       5         416     1    NA  30.5              0
##  2             3975       6         416     1    NA  30.7              0
##  3             4324       9         452     0     2  42.0              0
##  4             4325      10         452     0     2  42.3              0
##  5             4326      11         452     0     2  42.5              1
##  6             4327      12         452     0     2  42.8              0
##  7             4411       6         462     0     2  31.3              0
##  8             4412       7         462     0     2  31.5              0
##  9             4413       8         462     0     2  31.8              0
## 10             7712       1         806     0     3  47.6              0
## # ... with 26 more rows, and 3 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>
```

22

Notes:

Strings and Factors: The most frequent operation involving strings is to search for an exact match. You can use str_detect to find cases where the match exists (often useful in conjunction with filter), and you can use str_replace to replace an instance of a match with something else (often useful in conjunction with mutate)

str_detect(string_vec, "^i think") - Starts with i think str_detect(string_vec, "i think$") - ends with i think str_detect(string_vec,"[Bb]ush") str_detect(string_vec, "[1](a-zA-Z]") str_detect(string_vec, "7.11") The character . matches any character. Some characters are "special". These include [and], (and), and .. If you want to search for these, you have to indicate they're special using . Unfortunately, is also special, so things get weird. str_detect(string_vec, "\[") - search for an actual character i.e, the open bracket [

*imp: select(-contains"")

example for separate and str_replace:

data_marj = table_marj %>% select(-contains("P Value")) %>% # pivot_longer(-State, names_to = "age_year", values_to = "percent") %>%

separate(age_year, into = c("age", "year"), sep = "\(") %>% #seperating the age_year into age and year, seperated by an open parenthesis. sep = "\(" means separate from where there is an (. We have to give \ since (is a special character.

mutate(year = str_replace(year, "\)", ""), #After the above step, year had a) and we would want to remove that. We can do that by replacing that with a blank. str_replace(year,"\)","") is replacing) with a black. Again we used \ since) is a spl character. percent = str_replace(percent,"[a-c]","")$, $there is a character a, b, or c at the end of the percent values which we would want to remove. To remove, use $str_replace$, and $[a-c]$ shows the presence of a-c at the end of the percent string and we want to replace that with a blank. percent = as.numeric(percent)) %>% filter(!(State %in% c("Total U.S.", "Northeast", "Midwest", "South", "West")))

#Interactivity

practice_data %>% drop_na() %>% select(quarter, race, health_score) %>% mutate(race = factor(race)) %>% group_by(race, quarter) %>% summarise(mean_score = mean(health_score)) %>% plot_ly(x = ~quarter, y = ~ mean_score, color = ~race, type = "bar")

```
age_function = function(x){
  tibble(
    mean = mean(x),
    minimum = min(x),
    maximum = max(x)
  )
}
```

```
a = practice_data %>%
  select(
    age, salary, health_score
  )
```

```
age_function(x = a$age)
```

```
## # A tibble: 1 x 3
##     mean minimum maximum
##    <dbl>   <dbl>   <dbl>
## 1  30.6        7     172
```

[1]0-9

```
age_function(x = a$health_score)
```

```
## # A tibble: 1 x 3
##    mean minimum maximum
##   <dbl>   <dbl>   <dbl>
## 1  3.59   0.627      10
```

```
age_function(x = a$salary)
```

```
## # A tibble: 1 x 3
##     mean minimum maximum
##    <dbl>   <dbl>   <dbl>
## 1 48298.  28351.  68826.
```

```
practice_data2 = practice_data %>%
  filter(
    employee_id %in% 1:3
  ) %>%
  select(
    employee_id, quarter, age, hospital_visit, salary, health_score
  )
```

#Nesting

```
employee_nest = practice_data2 %>%
  nest(
    data = quarter:health_score
  ) #nesting the age,hosputal visit, salary and health score within the employee id and quarter. dATA i
```

Is the list column really a list? Let's check?

```
employee_nest %>%
  pull(data)
```

```
## [[1]]
## # A tibble: 12 x 5
##    quarter   age hospital_visit salary health_score
##      <dbl> <dbl>          <dbl>  <dbl>        <dbl>
## 1        1  27.3              0 36907.         3.70
## 2        2  27.8              0 37907.         4.98
## 3        3  28.1              0 38907.         4.01
## 4        4  28.3              0 39907.         2.34
## 5        5  28.6              0 40907.         2.11
## 6        6  28.8              0 41907.         1.46
## 7        7  29.1              0 42907.         4.73
## 8        8  29.3              0 43907.         2.34
## 9        9  29.6              0 44907.         2.76
## 10      10  29.8              0 45907.         2.83
## 11      11  30.1              0 46907.         1.80
## 12      12  30.3              0 47907.         1.70
##
```

```
## [[2]]
## # A tibble: 12 x 5
##    quarter   age hospital_visit salary health_score
##      <dbl> <dbl>          <dbl>  <dbl>        <dbl>
## 1        1  23.5              0 42050.         2.23
## 2        2  24.0              0 43050.         3.96
## 3        3  24.3              0 44050.         2.58
## 4        4  24.5              0 45050.         3.04
## 5        5  24.8              0 46050.         4.39
## 6        6  25.0              0 47050.         3.47
## 7        7  25.3              0 48050.         2.26
## 8        8  25.5              1 49050.         4.62
## 9        9  25.8              0 50050.         2.65
## 10      10  26.0              0 51050.         2.18
## 11      11  26.3              1 52050.         2.50
## 12      12  26.5              0 53050.         1.97
##
## [[3]]
## # A tibble: 7 x 5
##    quarter   age hospital_visit salary health_score
##      <dbl> <dbl>          <dbl>  <dbl>        <dbl>
## 1        6  36.8              0 45631.         3.00
## 2        7  37.0              0 46631.         3.01
## 3        8  37.3              0 47631.         3.63
## 4        9  37.5              0 48631.         2.92
## 5       10  37.8              0 49631.         2.59
## 6       11  38.0              0 50631.         3.91
## 7       12  38.3              1 51631.         3.49
```

```
employee_nest$data[[1]]
```

```
## # A tibble: 12 x 5
##    quarter   age hospital_visit salary health_score
##      <dbl> <dbl>          <dbl>  <dbl>        <dbl>
## 1        1  27.3              0 36907.         3.70
## 2        2  27.8              0 37907.         4.98
## 3        3  28.1              0 38907.         4.01
## 4        4  28.3              0 39907.         2.34
## 5        5  28.6              0 40907.         2.11
## 6        6  28.8              0 41907.         1.46
## 7        7  29.1              0 42907.         4.73
## 8        8  29.3              0 43907.         2.34
## 9        9  29.6              0 44907.         2.76
## 10      10  29.8              0 45907.         2.83
## 11      11  30.1              0 46907.         1.80
## 12      12  30.3              0 47907.         1.70
```

Unnesting

```
employee_nest %>%
  unnest() #back to the original data
```

```
## # A tibble: 31 x 6
```

```
##    employee_id quarter   age hospital_visit salary health_score
##          <dbl>   <dbl> <dbl>          <dbl>  <dbl>        <dbl>
##  1           1       1   27.3              0 36907.         3.70
##  2           1       2   27.8              0 37907.         4.98
##  3           1       3   28.1              0 38907.         4.01
##  4           1       4   28.3              0 39907.         2.34
##  5           1       5   28.6              0 40907.         2.11
##  6           1       6   28.8              0 41907.         1.46
##  7           1       7   29.1              0 42907.         4.73
##  8           1       8   29.3              0 43907.         2.34
##  9           1       9   29.6              0 44907.         2.76
## 10           1      10   29.8              0 45907.         2.83
## # ... with 21 more rows
```

Trying useful things with the list column:

```
employee_1 = employee_nest$data[[1]]
lm(health_score ~ age, data = employee_1)
```

```
##
## Call:
## lm(formula = health_score ~ age, data = employee_1)
##
## Coefficients:
## (Intercept)          age
##     23.1901      -0.7016
```

```
employee_2 = employee_nest$data[[2]]
lm(health_score ~ age, data = employee_2)
```

```
##
## Call:
## lm(formula = health_score ~ age, data = employee_2)
##
## Coefficients:
## (Intercept)          age
##       9.375       -0.254
```

Instead of repeating it again and again for all the three employees, we could:

```
lm(health_score ~ age, data = employee_nest$data[[1]])
```

```
##
## Call:
## lm(formula = health_score ~ age, data = employee_nest$data[[1]])
##
## Coefficients:
## (Intercept)          age
##     23.1901      -0.7016
```

```r
lm(health_score ~ age, data = employee_nest$data[[2]])
```

```
## 
## Call:
## lm(formula = health_score ~ age, data = employee_nest$data[[2]])
## 
## Coefficients:
## (Intercept)          age
##       9.375       -0.254
```

We could use a loop:

```r
output = vector("list", length = 3)

for (i in 1:3) {
  output[[i]] = lm(health_score ~ age, data = employee_nest$data[[i]])
}
```

```r
lm_fun = function(x) {
  lm(health_score ~ age, data = x)
} #written a function for the linear model
```

```r
output = vector("list", length = 3)

for (i in 1:3) {
  output[[i]] = lm_fun(x = employee_nest$data[[i]])
}
```

Use map:

```r
output = map(employee_nest$data, lm_fun)
```

Linear models:

```r
fit = lm(health_score ~ age + salary + hospital_visit + sex + race, data = practice_data )
summary(fit)
```

```
## 
## Call:
## lm(formula = health_score ~ age + salary + hospital_visit + sex +
##     race, data = practice_data)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.7863 -1.1738 -0.4700  0.4878  7.0869
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.025e+00  1.741e-01  11.626  < 2e-16 ***
## age          4.414e-02  2.061e-03  21.422  < 2e-16 ***
## salary       6.635e-07  3.249e-06   0.204 0.838195
```

```
## hospital_visit  7.925e-01  4.642e-02  17.070  < 2e-16 ***
## sex             3.760e-01  3.383e-02  11.115  < 2e-16 ***
## race            -7.407e-02  2.068e-02  -3.582 0.000342 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.896 on 16921 degrees of freedom
##   (2176 observations deleted due to missingness)
## Multiple R-squared:  0.05232,    Adjusted R-squared:  0.05204
## F-statistic: 186.8 on 5 and 16921 DF,  p-value: < 2.2e-16
```

```
coef(fit)
```

```
##    (Intercept)           age        salary hospital_visit            sex
##   2.024509e+00   4.414295e-02   6.635303e-07   7.924508e-01   3.760376e-01
##           race
##  -7.406799e-02
```

```
summary(fit)$coef
```

```
##                    Estimate   Std. Error    t value      Pr(>|t|)
## (Intercept)    2.024509e+00 1.741438e-01 11.6255038  4.015324e-31
## age            4.414295e-02 2.060676e-03 21.4215849 1.810104e-100
## salary         6.635303e-07 3.249309e-06  0.2042066  8.381945e-01
## hospital_visit 7.924508e-01 4.642353e-02 17.0700228  8.648799e-65
## sex            3.760376e-01 3.383106e-02 11.1151576  1.329800e-28
## race          -7.406799e-02 2.067825e-02 -3.5819271  3.420256e-04
```

```
fit %>%
  broom::tidy() %>%
  knitr::kable(digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	2.025	0.174	11.626	0.000
age	0.044	0.002	21.422	0.000
salary	0.000	0.000	0.204	0.838
hospital_visit	0.792	0.046	17.070	0.000
sex	0.376	0.034	11.115	0.000
race	-0.074	0.021	-3.582	0.000

Diagnostics for linear models:

```
modelr::add_residuals(practice_data, fit)
```

```
## # A tibble: 19,103 x 11
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##              <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
## 1                1       1           1     1     0     3  27.3               0
## 2                2       2           2     1     0     3  27.8               0
```

```
## 3                   3        3          1     0     3  28.1              0
## 4                   4        4          1     0     3  28.3              0
## 5                   5        5          1     0     3  28.6              0
## 6                   6        6          1     0     3  28.8              0
## 7                   7        7          1     0     3  29.1              0
## 8                   8        8          1     0     3  29.3              0
## 9                   9        9          1     0     3  29.6              0
## 10                 10       10          1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, resid <dbl>
```

```r
modelr::add_predictions(practice_data, fit)
```

```
## # A tibble: 19,103 x 11
##     observation_num~ quarter employee_id   sex  race   age hospital_visit
##                <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
## 1   1                   1        1          1     0     3  27.3              0
## 2   2                   2        2          1     0     3  27.8              0
## 3   3                   3        3          1     0     3  28.1              0
## 4   4                   4        4          1     0     3  28.3              0
## 5   5                   5        5          1     0     3  28.6              0
## 6   6                   6        6          1     0     3  28.8              0
## 7   7                   7        7          1     0     3  29.1              0
## 8   8                   8        8          1     0     3  29.3              0
## 9   9                   9        9          1     0     3  29.6              0
## 10  10                 10       10          1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, pred <dbl>
```

Nesting

```r
fit_interaction = lm(health_score ~ sex + age + salary + race + age*sex + hospital_visit, data = practi
fit_interaction %>%
  broom::tidy()
```

```
## # A tibble: 7 x 5
##   term              estimate   std.error statistic  p.value
##   <chr>                <dbl>       <dbl>     <dbl>     <dbl>
## 1 (Intercept)      2.22        0.183          12.2   7.29e-34
## 2 sex             -0.0777      0.132          -0.587 5.57e- 1
## 3 age              0.0381      0.00267        14.3   4.72e-46
## 4 salary           0.000000362 0.00000325      0.112 9.11e- 1
## 5 race            -0.0716      0.0207         -3.46  5.36e- 4
## 6 hospital_visit   0.792       0.0464         17.1   8.30e-65
## 7 sex:age          0.0149      0.00420         3.54  3.95e- 4
```

Logistic regression:

```r
fit_logistic = glm(hospital_visit ~ age + salary, data = practice_data)
fit_logistic %>%
  broom::tidy() %>%
  mutate(
```

```
    OR = exp(estimate)
) %>%
select(
  term, estimate, p.value, OR
) %>%
knitr::kable(digits = 2)
```

term	estimate	p.value	OR
(Intercept)	0	0.94	1
age	0	0.19	1
salary	0	0.00	1

```
train_data = sample_frac(practice_data, size = 0.8 )
test_data = anti_join(practice_data, train_data, by = "observation_number")
```
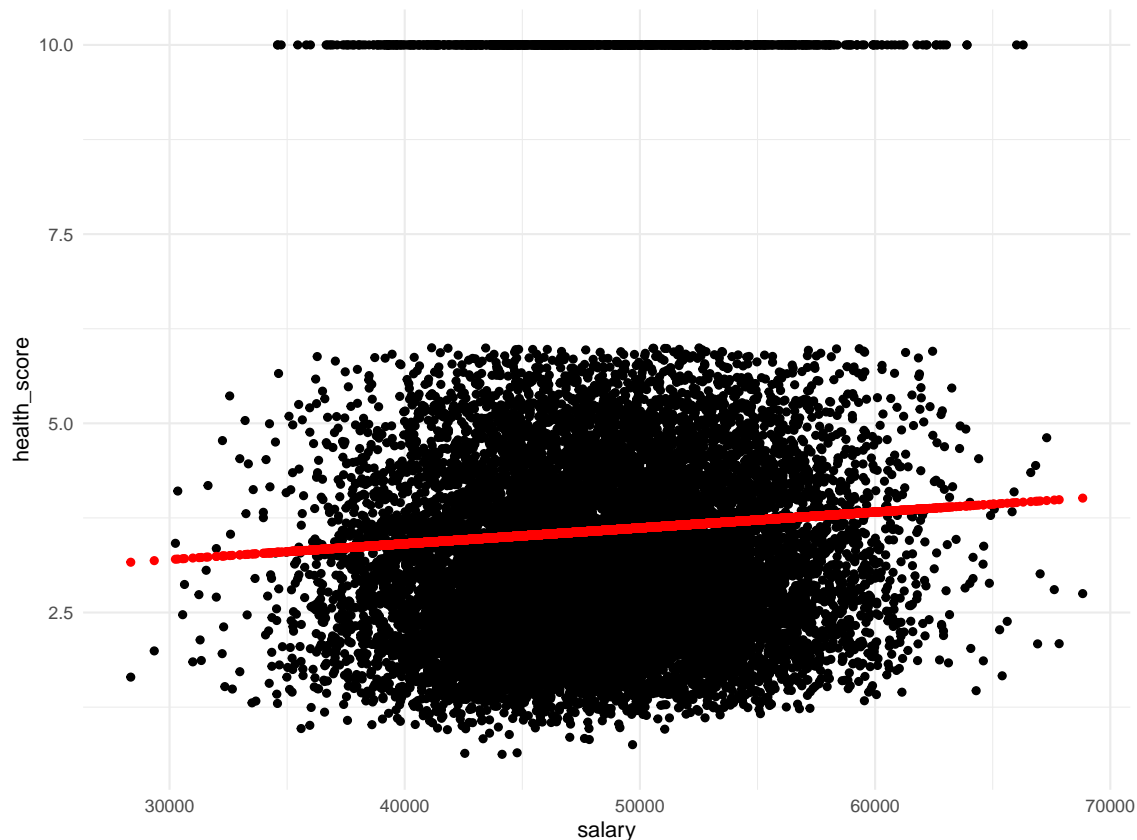
```
lm1 = lm(health_score ~ salary, data = train_data)
smooth_mod = mgcv::gam(health_score ~ s(salary), data = train_data)
wiggly_mod = mgcv::gam(health_score ~ s(salary, k = 30), sp = 10e-6, data = train_data)
```

```
train_data %>%
  add_predictions(lm1) %>%
  ggplot(
    aes(x = salary, y = health_score)
  ) + geom_point() + geom_point(aes(y = pred), color = "red")
```

```r
rmse(lm1, test_data)
```

```
## [1] 2.014703
```

Using modelr package for cross validation.

```r
cv_df = crossv_mc(practice_data, 100)
```

```r
cv_df %>%
  pull(train) %>%
  .[[1]] %>%
  as.tibble()
```

```
## # A tibble: 15,282 x 10
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##              <dbl>   <dbl>      <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1                1       1          1     1     0     3  27.3              0
##  2                2       2          2     1     0     3  27.8              0
##  3                3       3          3     1     0     3  28.1              0
##  4                4       4          4     1     0     3  28.3              0
##  5                5       5          5     1     0     3  28.6              0
##  6                6       6          6     1     0     3  28.8              0
##  7                7       7          7     1     0     3  29.1              0
##  8                8       8          8     1     0     3  29.3              0
##  9                9       9          9     1     0     3  29.6              0
## 10               10      10         10     1     0     3  29.8              0
## # ... with 15,272 more rows, and 3 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>
```

```r
cv_df %>%
  pull(test) %>%
  .[[1]] %>%
  as.tibble()
```

```
## # A tibble: 3,821 x 10
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##              <dbl>   <dbl>      <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1               11      11          1     0     3  30.1                  0
##  2               21       9          2     0     2  25.8                  0
##  3               30      11          3     1     1  38.0                  0
##  4               34      10          4     0     3  49.6                  0
##  5               38       4          5     1     1  26.2                  0
##  6               39       5          5     1     1  26.4                  1
##  7               42       8          5     1     1  27.2                  0
##  8               46      12          5     1     1  28.2                  0
##  9               47       3          6     0     3  31.8                  0
## 10               51       7          6     0     3  32.8                  0
## # ... with 3,811 more rows, and 3 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>
```

```
cv_df = cv_df %>%
  mutate(
   train =  map(train, as.tibble),
   test = map(test, as.tibble)
  )
```

Fitting the linear model to all of these datasets. . .

```
cv_df %>%
  mutate(
    linear_mod = map(.x = train, ~lm(health_score ~ salary, data = .x)),
    rmse = map2_dbl(.x = linear_mod, .y = test, ~ rmse(.x,.y))
  )
```

```
## # A tibble: 100 x 5
##    train                 test                .id  linear_mod  rmse
##    <list>                <list>              <chr> <list>     <dbl>
##  1 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 001   <lm>        1.97
##  2 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 002   <lm>        1.96
##  3 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 003   <lm>        1.99
##  4 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 004   <lm>        1.97
##  5 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 005   <lm>        1.96
##  6 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 006   <lm>        1.94
##  7 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 007   <lm>        1.99
##  8 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 008   <lm>        2.04
##  9 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 009   <lm>        1.99
## 10 <tibble [15,282 x 10]> <tibble [3,821 x 10]> 010   <lm>        1.91
## # ... with 90 more rows
```

```
cv_df = crossv_mc(practice_data, 100)
```

```
cv_df %>%
mutate(
  linear_mod = map(.x = train, ~lm(health_score ~ salary, data = .x)),
  rmse = map2_dbl(.x = linear_mod, .y = test, ~rmse(.x,.y))
)
```

```
## # A tibble: 100 x 5
##    train      test       .id   linear_mod  rmse
##    <list>     <list>     <chr> <list>     <dbl>
##  1 <resample> <resample> 001   <lm>        1.93
##  2 <resample> <resample> 002   <lm>        1.99
##  3 <resample> <resample> 003   <lm>        1.95
##  4 <resample> <resample> 004   <lm>        2.01
##  5 <resample> <resample> 005   <lm>        1.98
##  6 <resample> <resample> 006   <lm>        1.97
##  7 <resample> <resample> 007   <lm>        1.96
##  8 <resample> <resample> 008   <lm>        1.95
##  9 <resample> <resample> 009   <lm>        1.93
## 10 <resample> <resample> 010   <lm>        1.99
## # ... with 90 more rows
```