Dplyr 101

Fabrice Rossi

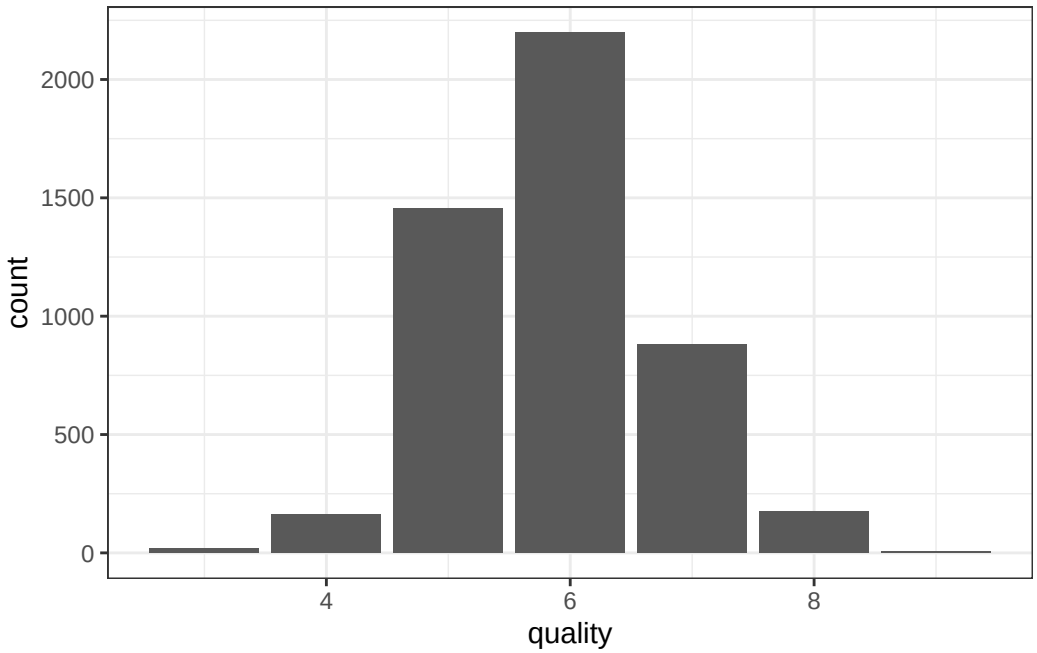
```
here::i_am("git-101-2024.Rproj")
library(here)
library(vroom)
library(ggplot2)
theme_set(theme_bw())
```

Data import

```
white_wine <- vroom(here("data", "winequality-white.csv"))
```

Number of observations	4898
Number of variables	12

```
ggplot(white_wine, aes(x = quality)) +
  geom_bar()
```



Data transformation

```
library(dplyr)
```

Column extraction

```
class(white_wine)
```

```
[1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
## class + column extraction
class(white_wine$quality)
```

```
[1] "numeric"
```

```
## column extraction and then class
white_wine$quality |> class()
```

```
[1] "numeric"
```

```
white_wine[["quality"]] |> class()
```

```
[1] "numeric"
```

```
white_wine$`fixed acidity` |> class()
```

```
[1] "numeric"
```

```
white_wine[["fixed acidity"]] |> class()
```

```
[1] "numeric"
```

```
## dplyr style
white_wine |>
  pull(quality) |>
  class()
```

```
[1] "numeric"
```

```
class(pull(white_wine, quality))
```

```
[1] "numeric"
```

With `dplyr`, `pull()` is the column extraction function, somewhat similar to the dollar `$` operator and to the double bracket `[[]]`.

Column recoding

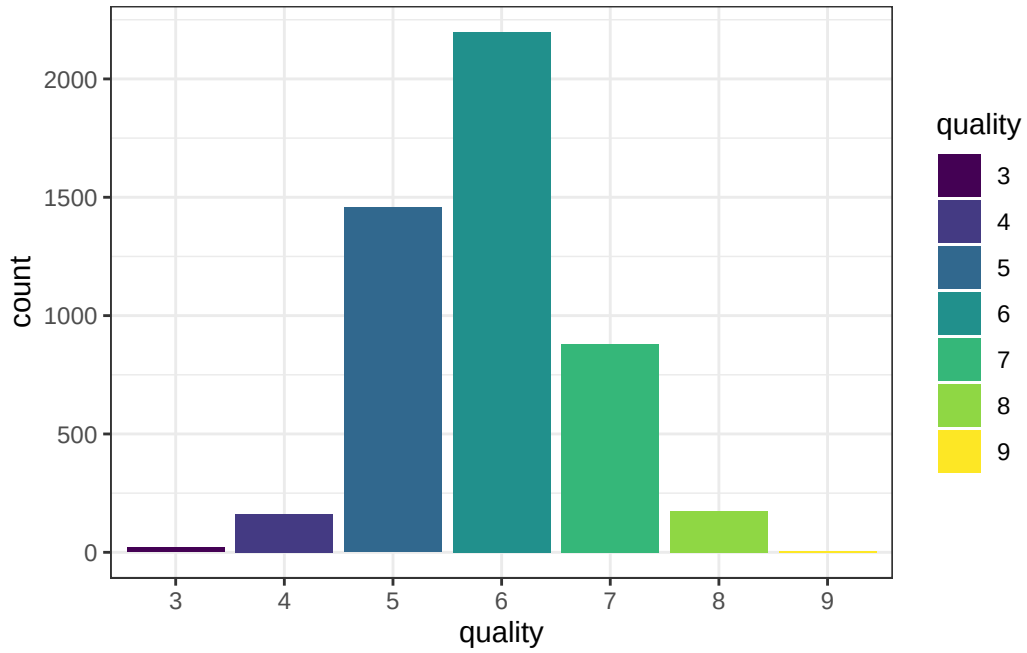
This ggplot call gives a warning because `quality` is numeric which is not supported for bar filling in `geom_bar()`.

```
ggplot(white_wine, aes(x = quality, fill = quality)) +
  geom_bar()
```

```r
white_wine$quality <- factor(white_wine$quality,
  ordered = TRUE,
  levels = 3:9
) ## 3:9 means 3, 4, 5, ..., 9
white_wine$quality <- white_wine$quality |> factor(
  ordered = TRUE,
  levels = 3:9
) ## 3:9 means 3, 4, 5, ..., 9
```

```r
white_wine <- white_wine |>
  mutate(quality = factor(quality, ordered = TRUE, levels = 3:9))
```

```r
ggplot(white_wine, aes(x = quality, fill = quality)) +
  geom_bar()
```



Computing new columns

This is again a task for `mutate()`.

```r
white_wine <- white_wine |>
  mutate(`captured sulfur dioxide` = `total sulfur dioxide` - `free sulfur dioxide`,
         `free sd %` = round(`free sulfur dioxide` / `total sulfur dioxide` * 100, 2))
```

Sub-setting

Selecting columns

Column sub-setting is done with the `select()` function.

```r
white_wine |>
  select(`fixed acidity`, `volatile acidity`, `citric acid`, pH)
```

```
# A tibble: 4,898 x 4
   `fixed acidity` `volatile acidity` `citric acid`    pH
             <dbl>              <dbl>         <dbl> <dbl>
 1             7                0.27          0.36  3
 2             6.3              0.3           0.34  3.3
 3             8.1              0.28          0.4   3.26
 4             7.2              0.23          0.32  3.19
 5             7.2              0.23          0.32  3.19
 6             8.1              0.28          0.4   3.26
 7             6.2              0.32          0.16  3.18
 8             7                0.27          0.36  3
 9             6.3              0.3           0.34  3.3
10             8.1              0.22          0.43  3.22
# i 4,888 more rows
```

Side note: we can display nicely data frames with the `knitr::kable()` function but the original data frame is too big, so we select the first 10 rows with `slice()`.

```r
white_wine |>
  select(`fixed acidity`, `volatile acidity`, `citric acid`, pH) |>
  slice(1:10) |>
  knitr::kable()
```

fixed acidity	volatile acidity	citric acid	pH
7.0	0.27	0.36	3.00
6.3	0.30	0.34	3.30

fixed acidity	volatile acidity	citric acid	pH
8.1	0.28	0.40	3.26
7.2	0.23	0.32	3.19
7.2	0.23	0.32	3.19
8.1	0.28	0.40	3.26
6.2	0.32	0.16	3.18
7.0	0.27	0.36	3.00
6.3	0.30	0.34	3.30
8.1	0.22	0.43	3.22

Positional selection based on the indices of the columns.

```
white_wine |>
  select(1:3)
```

Columns can also be selected based on conditions on their names or on their nature.

```
white_wine |>
  select(contains("acid") | pH)
```

Selection based on the content.

```
white_wine |>
  select(where(is.factor))
```

```
white_wine |>
  select(where(\(x) is.numeric(x) & (max(x) < 5)))
```

Selecting rows

Simple selection is based on indices with `slice()`.

```
white_wine |>
  slice(1:5, 150:155) |>
  select(alcohol, quality) |>
  knitr::kable()
```

alcohol	quality
8.8	6
9.5	6
10.1	6
9.9	6
9.9	6
9.4	6
11.5	7
9.9	6
9.4	6
10.4	5
9.7	6

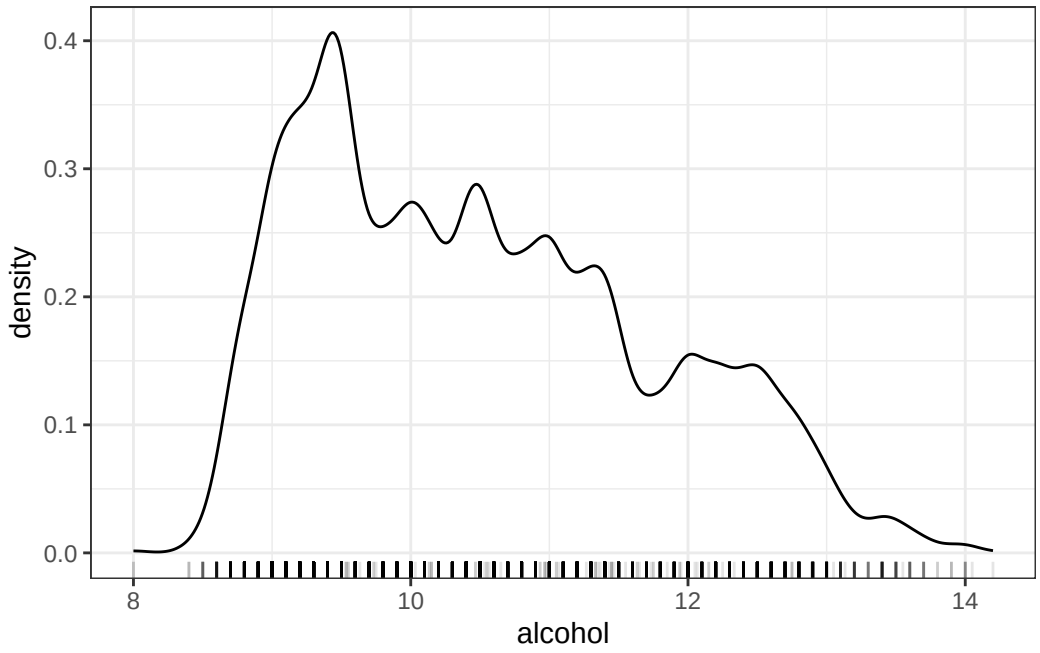
Condition based selection uses `filter()`.

```
white_wine |>
  filter(alcohol > 14) |>
  select(alcohol, quality, `citric acid`, `residual sugar`) |>
  knitr::kable()
```

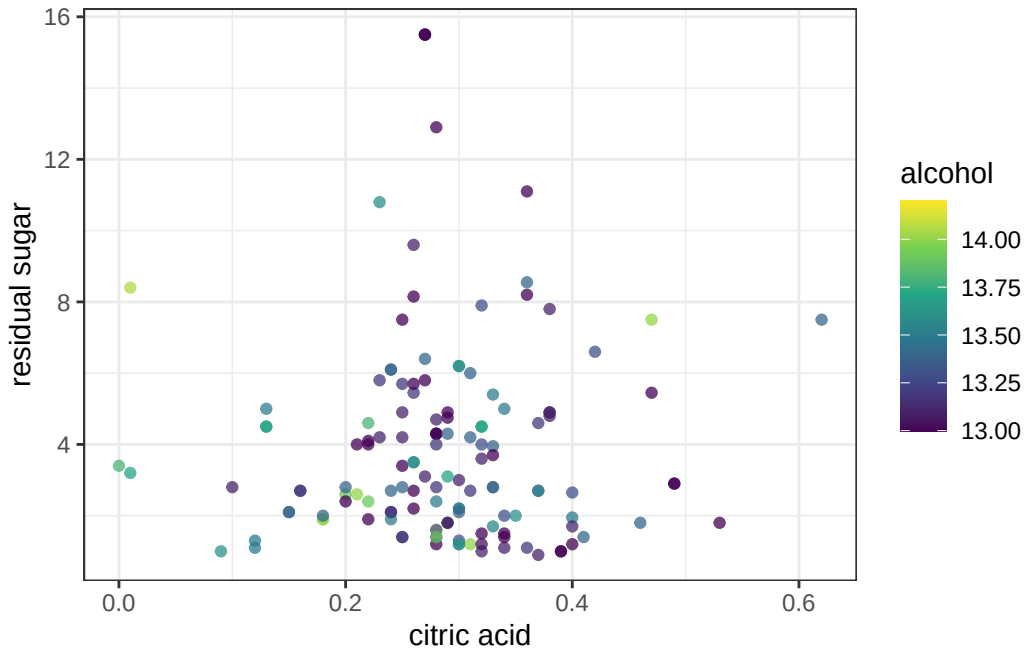
alcohol	quality	citric acid	residual sugar
14.20	7	0.28	1.6
14.05	7	0.01	8.4

Simple application

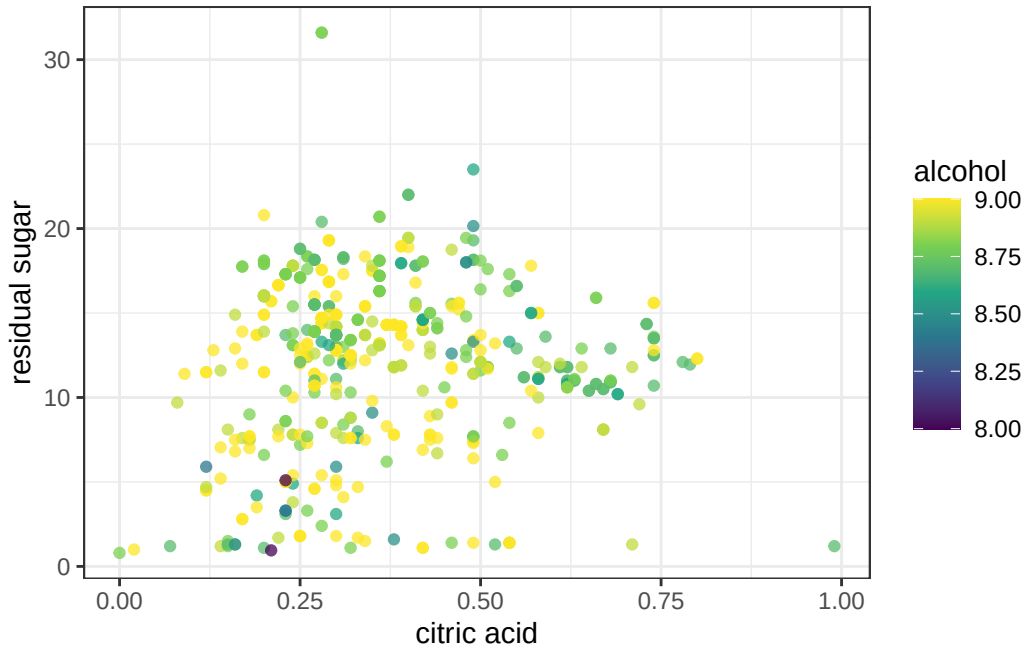
```
ggplot(white_wine, aes(x=alcohol)) +
  geom_density(bw="sj") +
  geom_rug(alpha = 0.1)
```



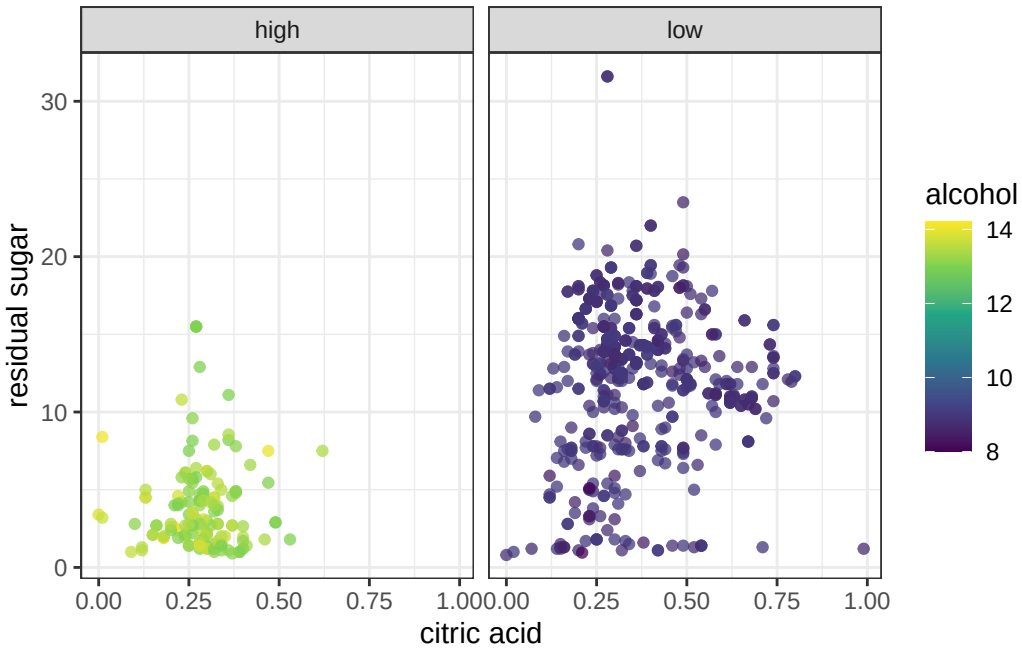

```
white_wine |>
  filter(alcohol >= 13, `citric acid` < 1) |>
  ggplot(aes(x=`citric acid`, y=`residual sugar`, color = alcohol)) +
  geom_point(alpha=0.75) +
  scale_color_viridis_c()
```




```
white_wine |>
  filter(alcohol <= 9) |>
  ggplot(aes(x=`citric acid`, y=`residual sugar`, color = alcohol)) +
  geom_point(alpha=0.75) +
  scale_color_viridis_c()
```




```
white_wine |>
  mutate(`alcohol category` = case_when(alcohol <= 9 ~ "low",
                                        alcohol >= 13 ~ "high",
                                        .default = "medium")) |>
  filter(`alcohol category` != "medium") |>
  filter(`citric acid` <= 1) |>
  ggplot(aes(x=`citric acid`, y=`residual sugar`, color = alcohol)) +
  geom_point(alpha=0.75) +
  scale_color_viridis_c() +
  facet_wrap(vars(`alcohol category`))
```



Aggregation functions

AKA summary functions: turn a possibly long vector into a single value (still a vector!)

- standard statistics: `max()`, `min()`, `median()`, etc.
- logical operations: `any()` and `all()`
- counts and ranks: `n()`

Simple summary use

Based on the `summarise()` function or inside a `mutate()` call.

```
white_wine |>
  summarise(
    med_alcohol = median(alcohol),
    sd(alcohol))
```

```
# A tibble: 1 x 2
  med_alcohol `sd(alcohol)`
        <dbl>         <dbl>
1        10.4          1.23
```

```
white_wine |>
  select(alcohol) |>
  mutate(m_a = median(alcohol))
```

```
# A tibble: 4,898 x 2
   alcohol   m_a
     <dbl> <dbl>
 1     8.8  10.4
 2     9.5  10.4
 3    10.1  10.4
 4     9.9  10.4
 5     9.9  10.4
 6    10.1  10.4
 7     9.6  10.4
 8     8.8  10.4
 9     9.5  10.4
10    11    10.4
# i 4,888 more rows
```

```
white_wine |>
  select(alcohol) |>
  mutate(a_m_median = alcohol - median(alcohol),
         .keep = "none")
```

```
# A tibble: 4,898 x 1
   a_m_median
        <dbl>
 1     -1.6
 2     -0.9
 3     -0.300
 4     -0.5
 5     -0.5
 6     -0.300
 7     -0.800
 8     -1.6
 9     -0.9
10      0.600
# i 4,888 more rows
```

```r
white_wine |>
  select(alcohol) |>
  mutate(na = (alcohol - mean(alcohol))/sd(alcohol),
         na_r = (alcohol - median(alcohol))/IQR(alcohol),
         .keep = "none")
```

```
# A tibble: 4,898 x 2
       na   na_r
    <dbl>  <dbl>
 1 -1.39  -0.842
 2 -0.824 -0.474
 3 -0.337 -0.158
 4 -0.499 -0.263
 5 -0.499 -0.263
 6 -0.337 -0.158
 7 -0.743 -0.421
 8 -1.39  -0.842
 9 -0.824 -0.474
10  0.395  0.316
# i 4,888 more rows
```

```r
white_wine |>
  summarise(
    mean(alcohol),
    sd(alcohol),
    median(alcohol),
    IQR(alcohol))
```

```
# A tibble: 1 x 4
  `mean(alcohol)` `sd(alcohol)` `median(alcohol)` `IQR(alcohol)`
            <dbl>         <dbl>             <dbl>          <dbl>
1            10.5          1.23              10.4            1.9
```

More complex summary functions

Essentially `quantile()`.

```r
quantile(white_wine$`residual sugar`)
```

```
  0%  25%  50%  75% 100%
 0.6  1.7  5.2  9.9 65.8
```

```
quantile(white_wine$`residual sugar`, probs = seq(0, 1, 0.1))
```

```
  0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
 0.6  1.2  1.5  2.0  3.4  5.2  7.0  8.6 11.2 14.0 65.8
```

```
mean(white_wine$`residual sugar`)
```

```
[1] 6.391415
```

```
white_wine |>
  summarise(quantile(`residual sugar`))
```

```
Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
dplyr 1.1.0.
i Please use `reframe()` instead.
i When switching from `summarise()` to `reframe()`, remember that `reframe()`
  always returns an ungrouped data frame and adjust accordingly.

# A tibble: 5 x 1
  `quantile(\`residual sugar\`)`
                           <dbl>
1                            0.6
2                            1.7
3                            5.2
4                            9.9
5                           65.8
```

```
white_wine |>
  reframe(quantile(`residual sugar`))
```

```
# A tibble: 5 x 1
  `quantile(\`residual sugar\`)`
                           <dbl>
1                            0.6
2                            1.7
3                            5.2
4                            9.9
5                           65.8
```

```
white_wine |>
  reframe((quantile(`residual sugar`)), mean(`residual sugar`))
```

```
# A tibble: 5 x 2
  `(quantile(\`residual sugar\`))` `mean(\`residual sugar\`)`
                             <dbl>                      <dbl>
1                              0.6                       6.39
2                              1.7                       6.39
3                              5.2                       6.39
4                              9.9                       6.39
5                             65.8                       6.39
```

To not use reframe:

```
sugar_stats <- white_wine |>
  reframe(list(quantile(`residual sugar`)),
          mean(`residual sugar`))
```

Group by

```
white_wine |>
  summarise(median(`residual sugar`),
            .by = quality)
```

```
# A tibble: 7 x 2
  quality `median(\`residual sugar\`)`
  <ord>                          <dbl>
1 6                               5.3
2 5                               7
3 7                               3.65
4 8                               4.3
5 4                               2.5
6 3                               4.6
7 9                               2.2
```

```
white_wine |>
  summarise(median(`residual sugar`),
            .by = quality) |>
  arrange(quality)
```

```
# A tibble: 7 x 2
  quality `median(\`residual sugar\`)`
  <ord>                          <dbl>
1 3                               4.6
2 4                               2.5
3 5                               7
4 6                               5.3
5 7                               3.65
6 8                               4.3
7 9                               2.2
```

```
white_wine |>
  summarise(median(`residual sugar`),
            n(),
            .by = quality) |>
  arrange(quality)
```

```
# A tibble: 7 x 3
  quality `median(\`residual sugar\`)` `n()`
  <ord>                          <dbl> <int>
1 3                               4.6     20
2 4                               2.5    163
3 5                               7     1457
4 6                               5.3   2198
5 7                               3.65   880
6 8                               4.3    175
7 9                               2.2      5
```

```
white_wine |>
  mutate(ha = alcohol >= 13 ) |> ## ha stands for high alcohol
  summarise(median(`residual sugar`),
            n(),
            .by = ha)
```

```
# A tibble: 2 x 3
  ha    `median(\`residual sugar\`)` `n()`
  <lgl>                         <dbl> <int>
1 FALSE                          5.3   4760
2 TRUE                           2.95   138
```

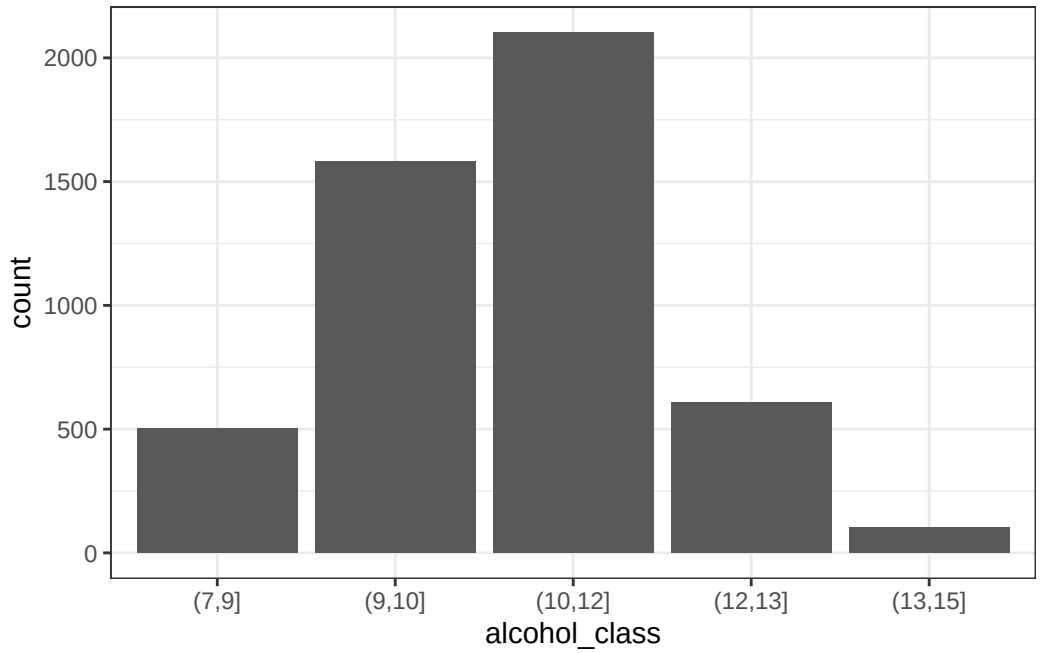
```
white_wine |>
  mutate(alcohol_class = cut(alcohol, breaks=c(7, 9, 10, 12, 13, 15)) ) |> ## breaks are cut
  summarise(median(`residual sugar`),
            n(),
            .by = alcohol_class)
```

```
# A tibble: 5 x 3
  alcohol_class `median(\`residual sugar\`)` `n()`
  <fct>                                <dbl> <int>
1 (7,9]                                 13.0   502
2 (9,10]                                 7.8  1583
3 (10,12]                                2.8  2102
4 (12,13]                                3     609
5 (13,15]                                2.8   102
```

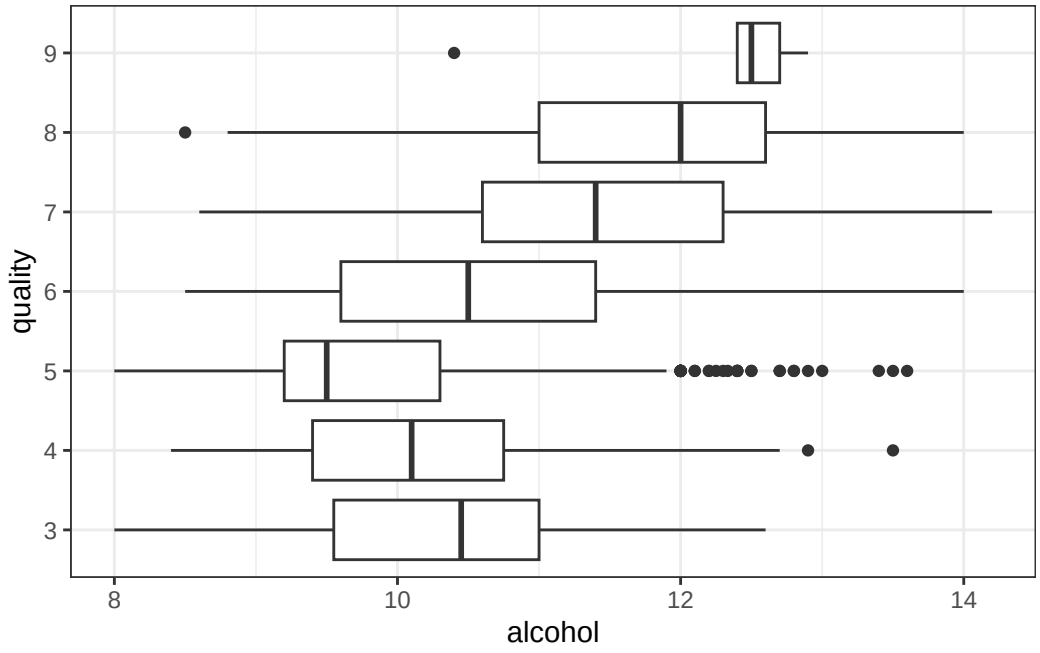
```
white_wine |>
  mutate(alcohol_class = cut(alcohol, breaks=c(7, 9, 10, 12, 13, 15),
                             ordered_results = TRUE)) |>
  summarise(median(`residual sugar`),
            n(),
            .by = alcohol_class)
```

```
# A tibble: 5 x 3
  alcohol_class `median(\`residual sugar\`)` `n()`
  <fct>                                <dbl> <int>
1 (7,9]                                 13.0   502
2 (9,10]                                 7.8  1583
3 (10,12]                                2.8  2102
4 (12,13]                                3     609
5 (13,15]                                2.8   102
```

```
white_wine |>
  mutate(alcohol_class = cut(alcohol, breaks=c(7, 9, 10, 12, 13, 15),
                             ordered_results = TRUE)) |>
  ggplot(aes(x=alcohol_class)) +
  geom_bar()
```




```
ggplot(white_wine,aes(x=alcohol,y=quality))+
  geom_boxplot()
```



```
white_wine |>
  mutate(alcohol_class = cut(alcohol, breaks=c(7, 9, 10, 12, 13, 15),
                             ordered_results = TRUE)) |>
  ggplot(aes(x=alcohol_class, fill=quality)) +
  geom_bar(position="fill")
```