03 - Data Visualisation

David Vichansky [6819516]

14-11-2020

Here is an example file you can write.

First, load the packages:

```
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

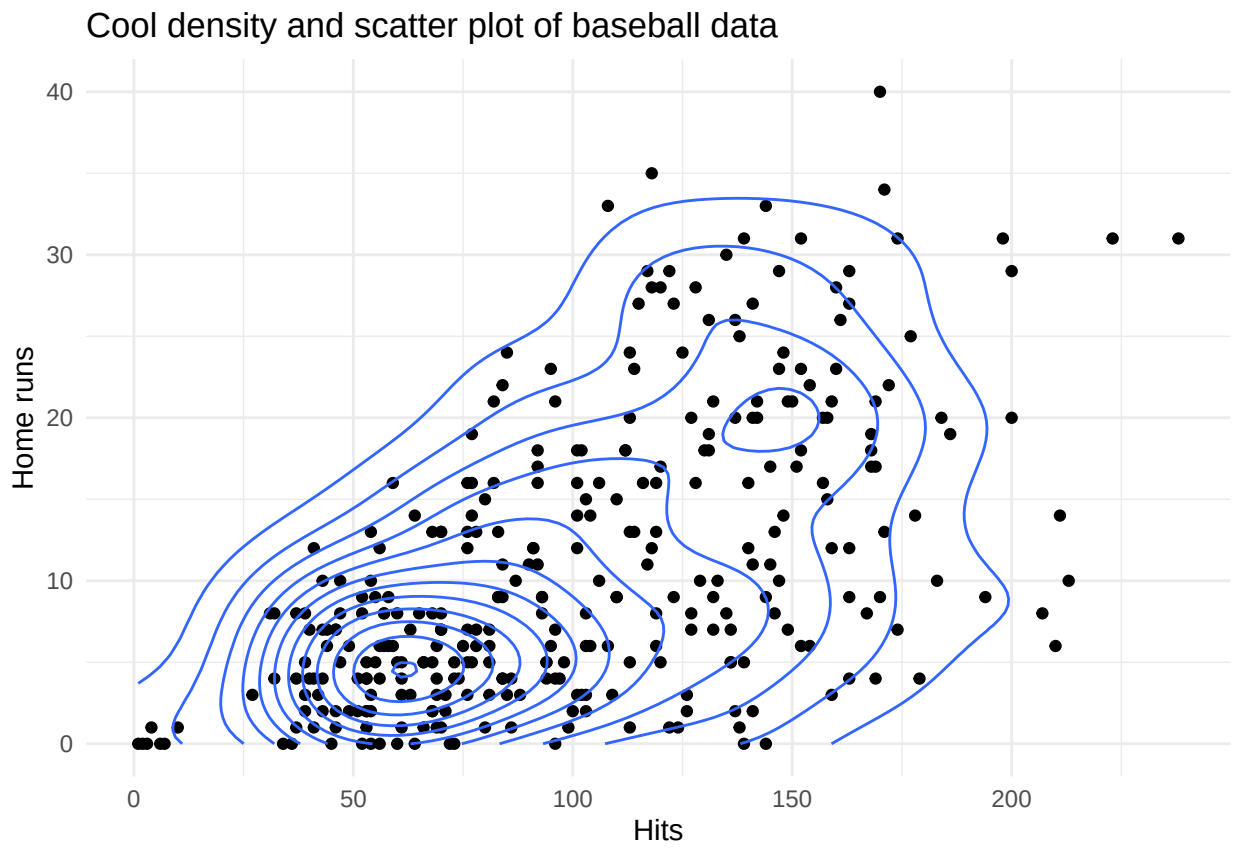
```
## -- Conflicts ------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(haven)
library(readxl)
library(tinytex)
library(ggplot2)
```

1. Name the aesthetics, geoms, scales, and facets of the above visualisation. Also name any statistical transformations or special coordinate systems.

```
#Hitters <- Hitters
homeruns_plot <-
  ggplot(Hitters, aes(x = Hits, y = HmRun)) +
  geom_point() +
  labs(x = "Hits", y = "Home runs")

homeruns_plot +
  geom_density_2d() +
  labs(title = "Cool density and scatter plot of baseball data") +
  theme_minimal()
```



2. Run the code below to generate data. There will be three vectors in your environment. Put them in a data frame for entering it in a ggplot() call using either the data.frame() or the tibble() function. Give informative names and make sure the types are correct (use the as.() functions). Name the result gg_students.

```
## code to run
set.seed(1234)
student_grade  <- rnorm(32, 7)
student_number <- round(runif(32) * 2e6 + 5e6)
programme      <- sample(c("Science", "Social Science"), 32, replace = TRUE)
##

gg_students <- data.frame(student_grade, student_number, programme) %>%
  `colnames<-`(c('grade', 'number', 'programme')) %>%
  mutate(grade=as.numeric(grade)) %>%
  mutate(number=as.numeric(number)) %>%
  mutate(programme=as.factor(programme))

# check 'class' types
lapply(gg_students, class)
```

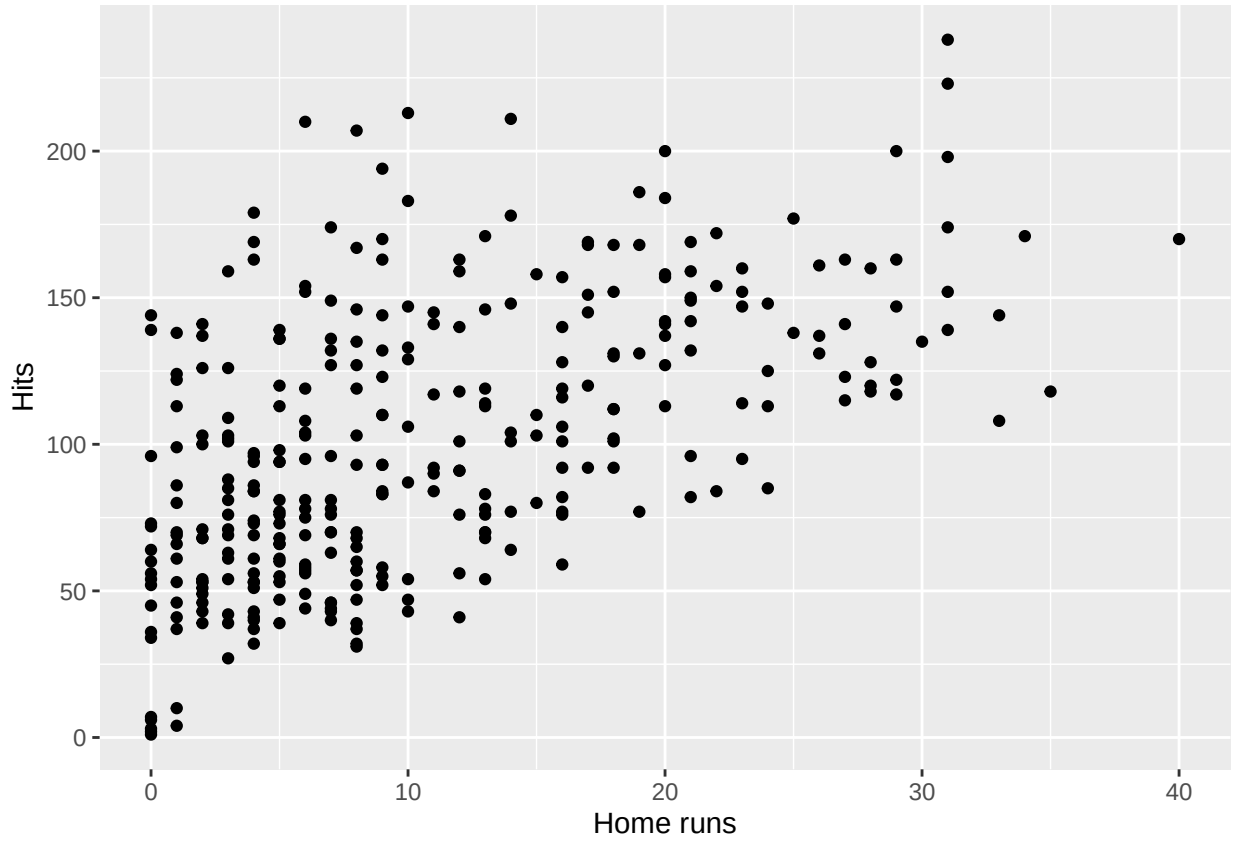
```
## $grade
```

```
## [1] "numeric"
##
## $number
## [1] "numeric"
##
## $programme
## [1] "factor"
```

3. Plot the first homeruns_plot again, but map the Hits to the y-axis and the HmRun to the x-axis instead.

```
homeruns_plot <-
  ggplot(Hitters, aes(x = HmRun, y = Hits)) +
  geom_point() +
    labs(x = "Home runs", y = "Hits")

homeruns_plot
```

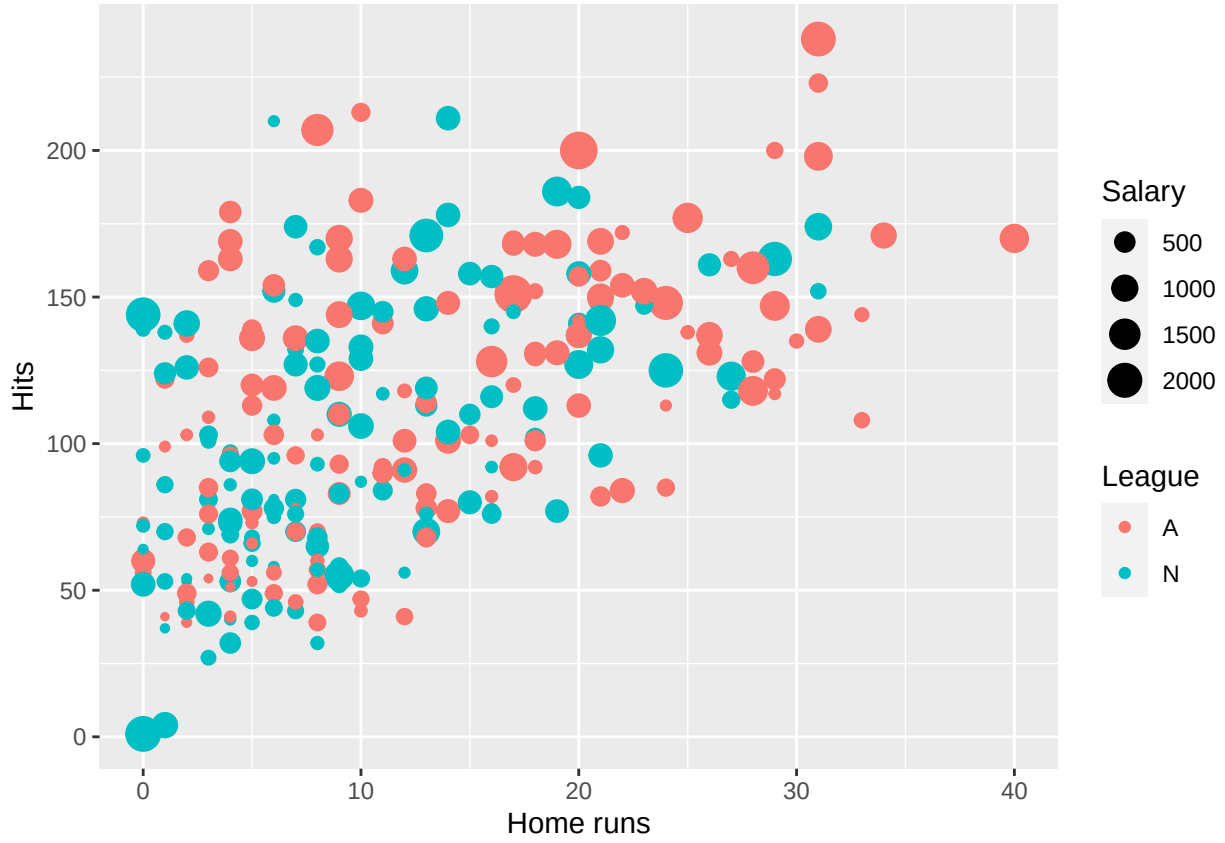


4. Recreate the same plot once more, but now also map the variable League to the colour aesthetic and the variable Salary to the size aesthetic.

```
ggplot(Hitters, aes(x = HmRun, y = Hits, color = League, size = Salary)) +
  geom_point() +
```

```
    labs(x = "Home runs", y = "Hits")
```

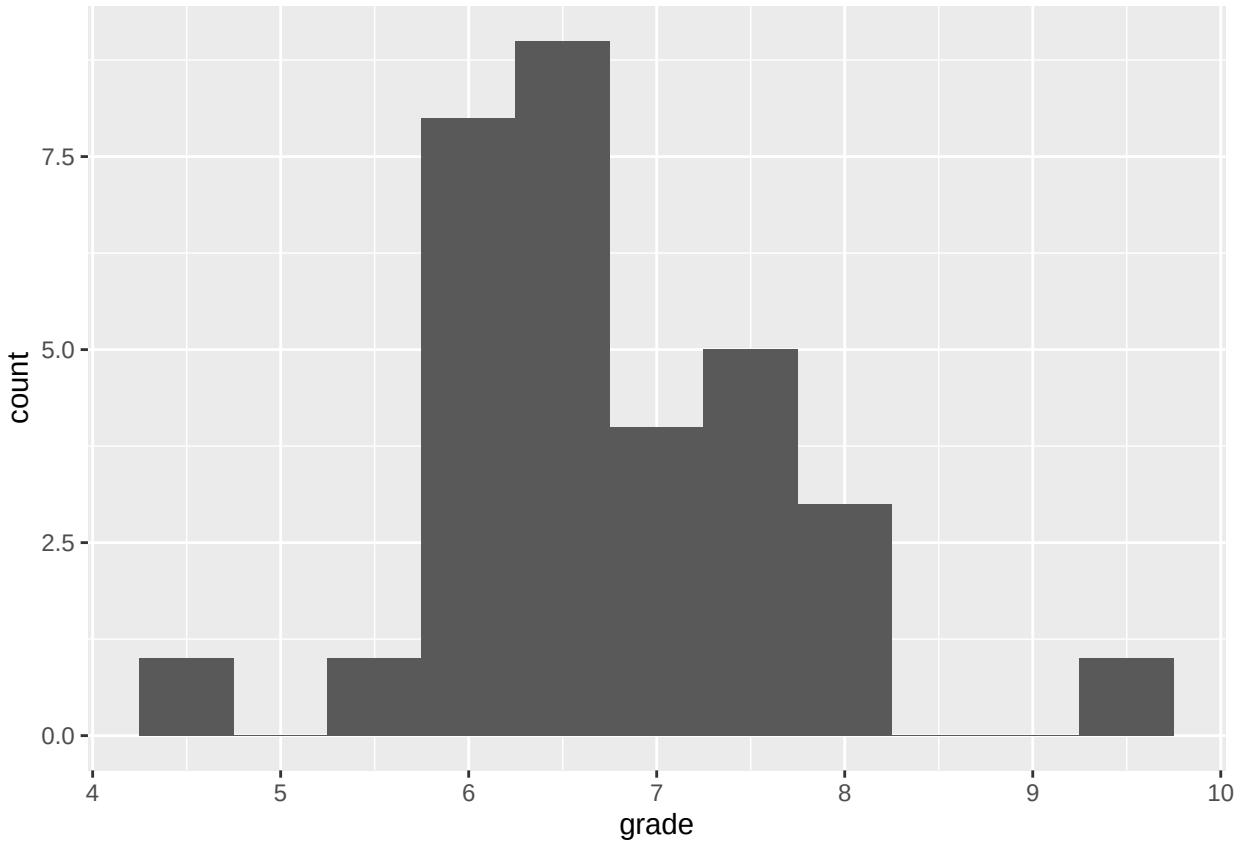
```
## Warning: Removed 59 rows containing missing values (geom_point).
```



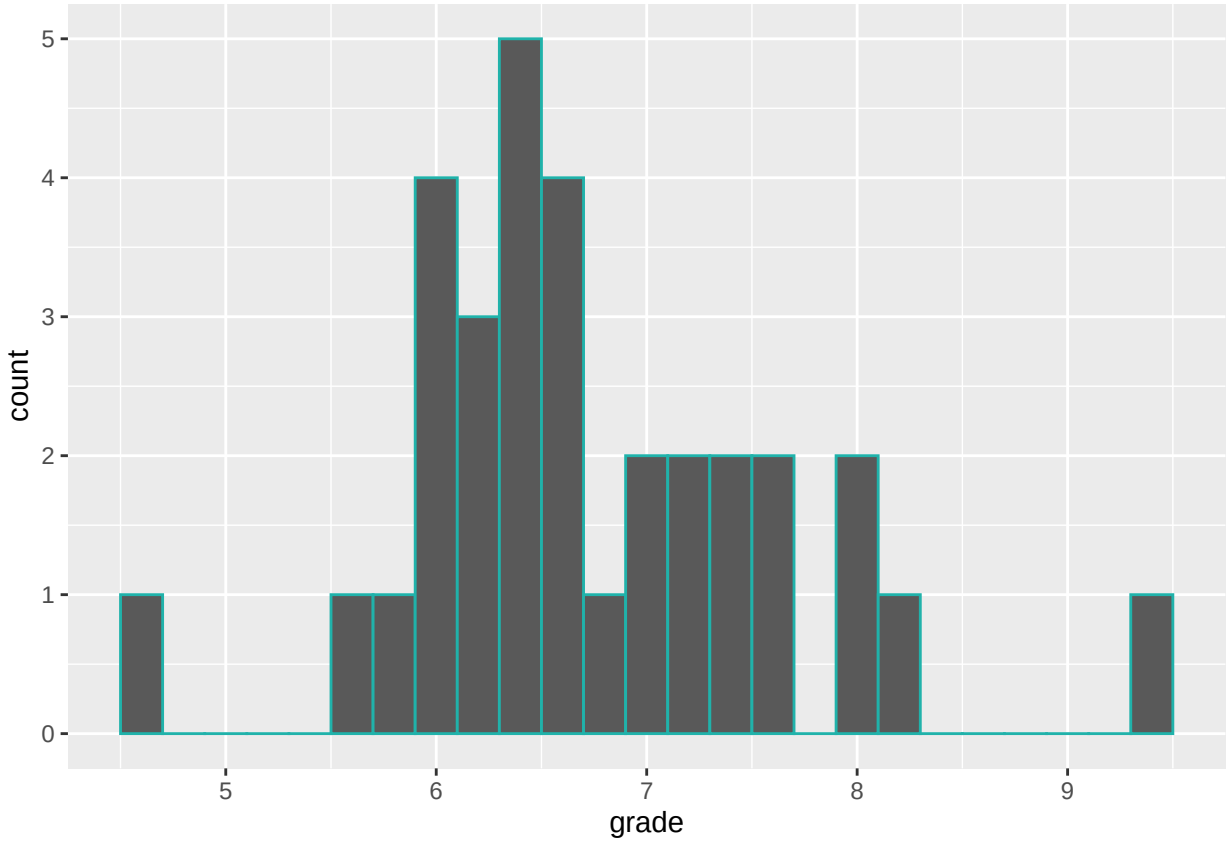
5. Look at the many different 'geoms'.

6. Use geom_histogram() to create a histogram of the grades of the students in the gg_students dataset. Play around with the binwidth argument of the geom_histogram() function.

```
ggplot(data = gg_students) +
geom_histogram(mapping = aes(x = grade), binwidth = 0.5)
```

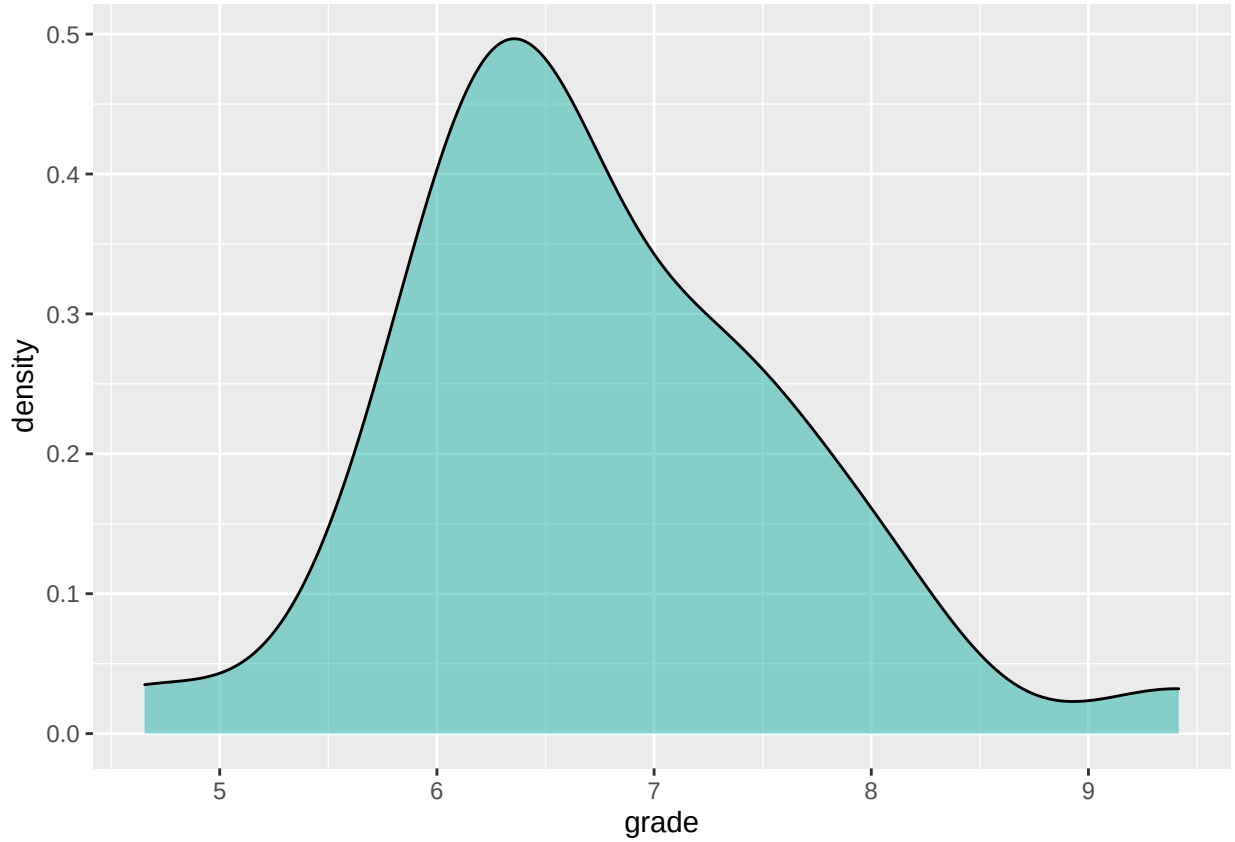



```
ggplot(data = gg_students) +
geom_histogram(mapping = aes(x = grade), binwidth = 0.2, color = "Light seagreen")
```



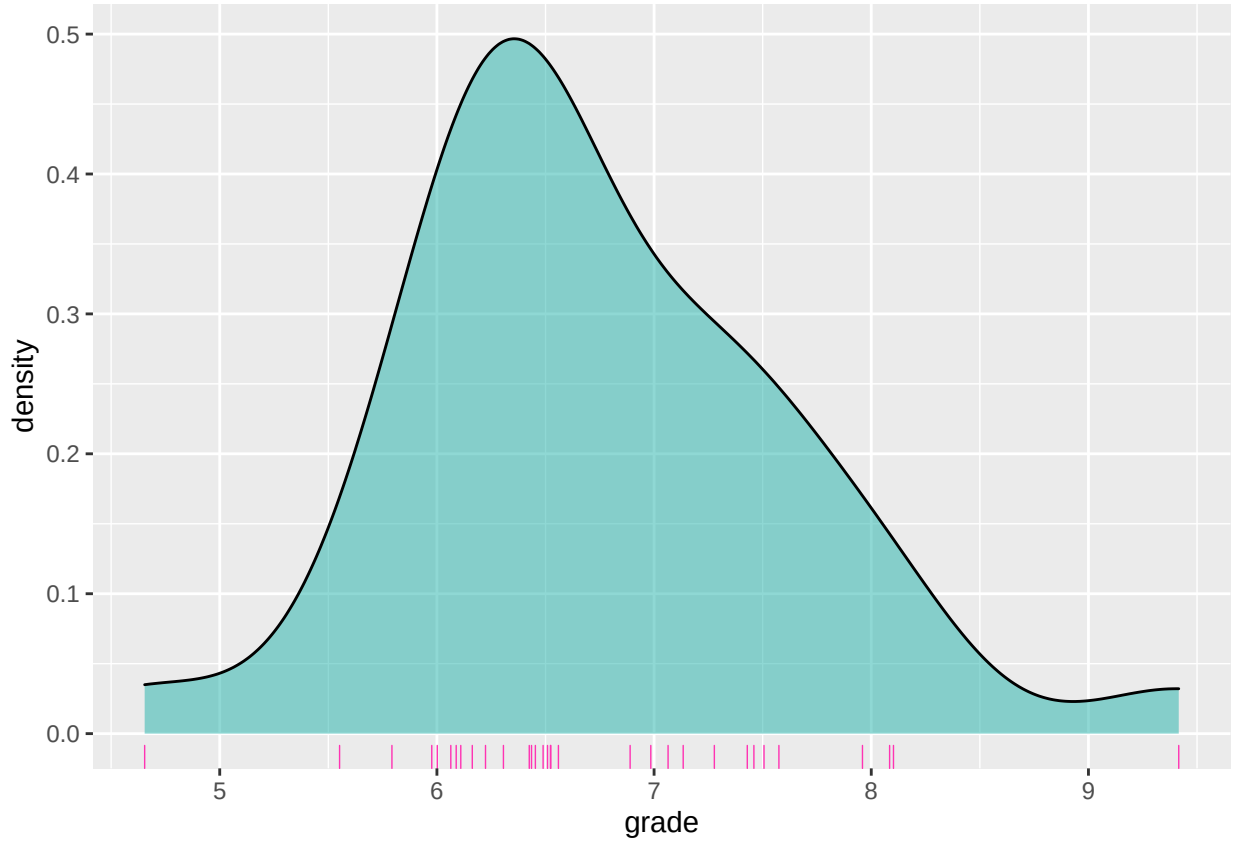
7. Use geom_density() to create a density plot of the grades of the students in the gg_students dataset. Add the argument fill = "light seagreen" to geom_density().

```
ggplot(gg_students, aes(grade)) +
  geom_density(fill = "Light Sea Green", alpha = 0.5)
```



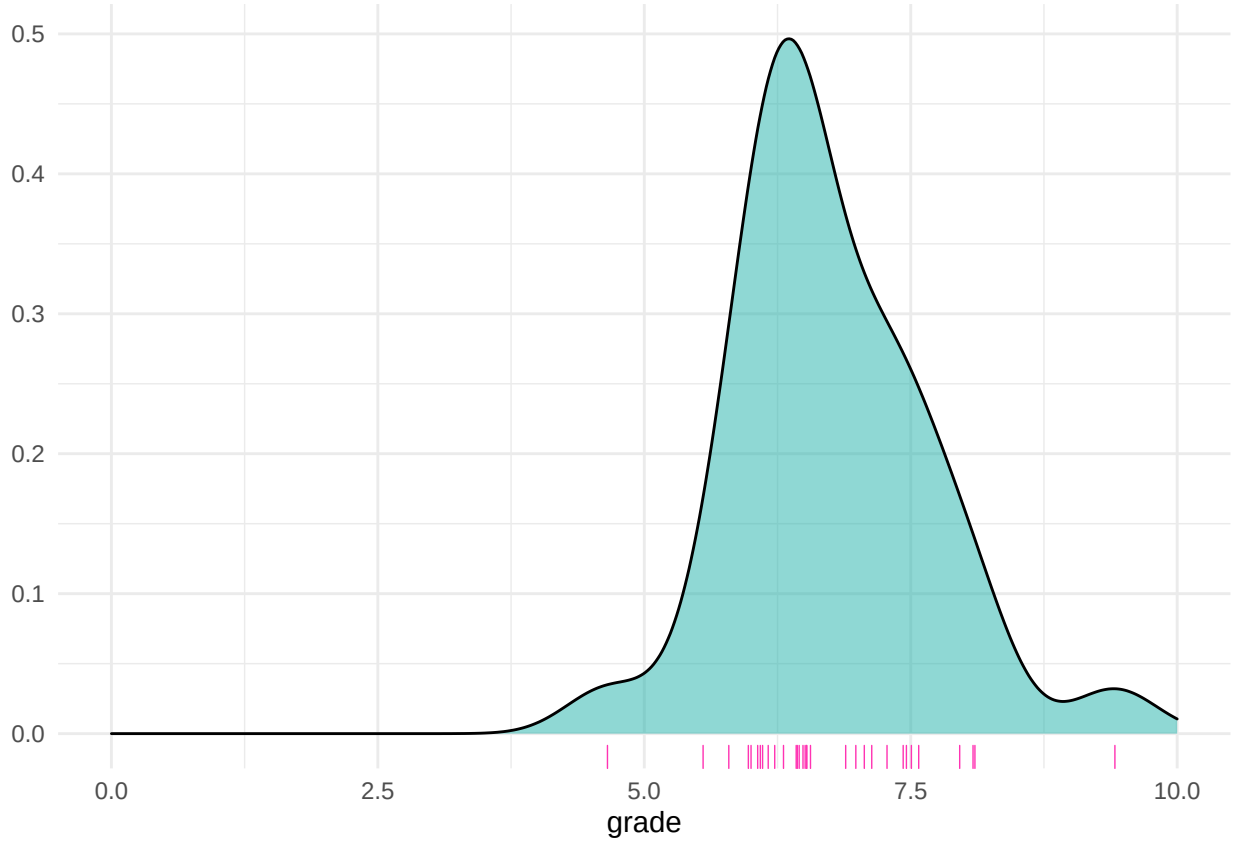
8. Add rug marks to the density plot through geom_rug(). You can edit the colour and size of the rug marks using those arguments within the geom_rug() function.

```
ggplot(gg_students, aes(grade)) +
  geom_density(fill = "Light Sea Green", alpha = 0.5) +
  geom_rug(color = "Maroon1", size = 0.25, outside = FALSE)
```



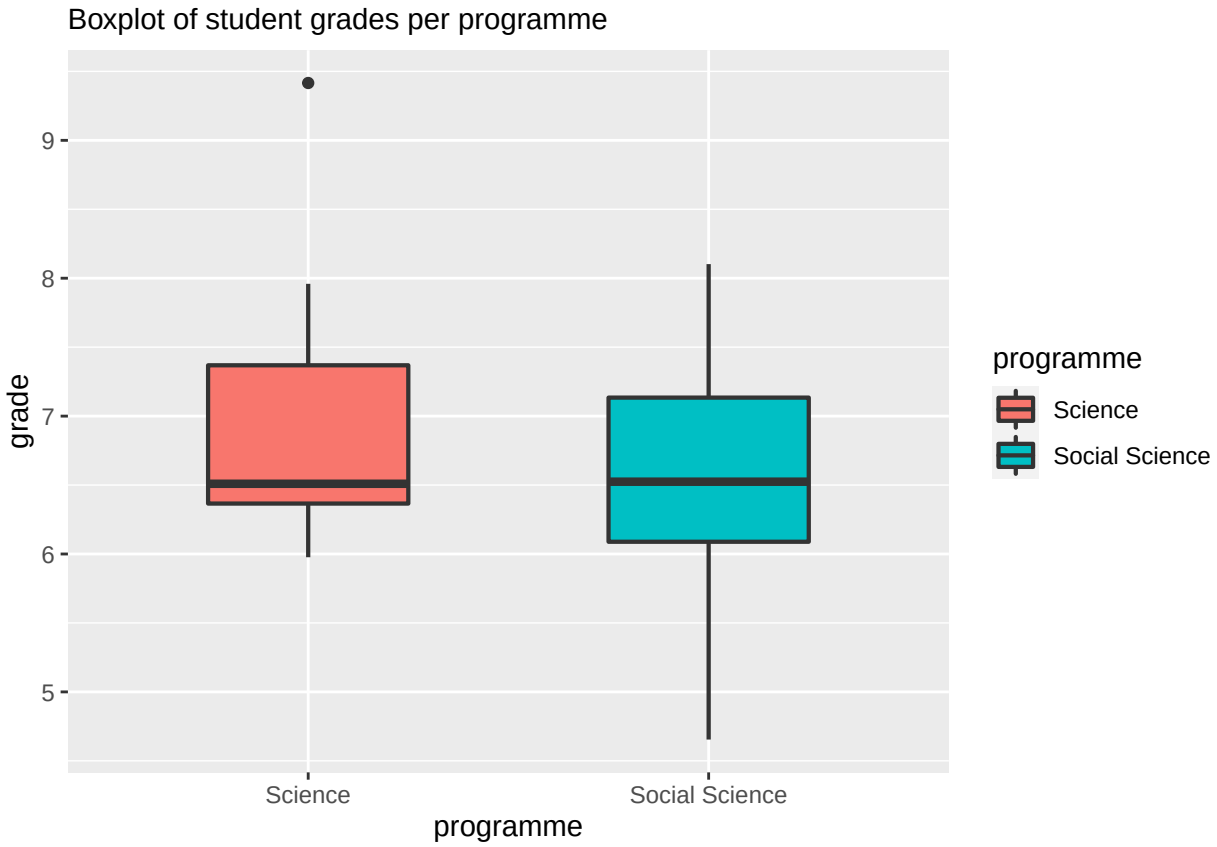
9. Increase the data to ink ratio by removing the y axis label, setting the theme to theme_minimal(), and removing the border of the density polygon. Also set the limits of the x-axis to go from 0 to 10 using the xlim() function, because those are the plausible values for a student grade.

```
ggplot(gg_students, aes(grade)) +
geom_density(fill = "Light Sea Green", alpha = 0.5) +
  geom_rug(color = "Maroon1", size = 0.25, outside = FALSE) +
  ylab(NULL) +
  xlim(0, 10) +
  theme_minimal(base_size = 11)
```



10. Create a boxplot of student grades per programme in the gg_students dataset you made earlier: map the programme variable to the x position and the grade to the y position. For extra visual aid, you can additionally map the programme variable to the fill aesthetic.

```
ggplot(gg_students, aes(x = programme, y = grade, fill = programme)) +
  geom_boxplot(width=0.5,lwd=0.75) +
  labs(subtitle="Boxplot of student grades per programme")
```

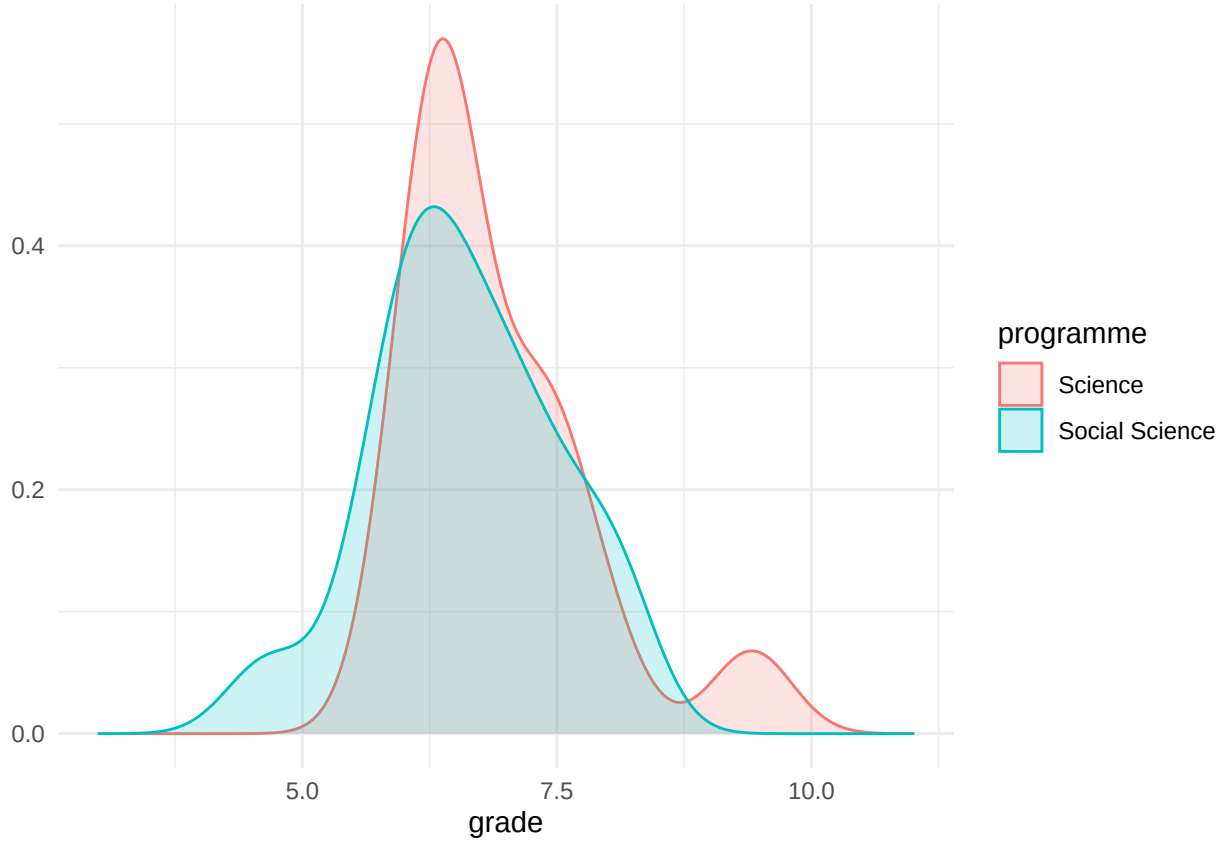


11. What do each of the horizontal lines in the boxplot mean? What do the vertical lines (whiskers) mean?

The box is bounded by "hinges" by that represent the quartiles Q3 and Q1 respectively, and with a horizontal median line through it. Each whisker is drawn out to the most extreme data point that is less than 1.5 IQR's beyond the corresponding "hinge". Therefore, the whisker ends correspond to the minimum and maximum values of the data excluding the outliers.

12. Comparison of distributions across categories can also be done by adding a fill aesthetic to the density plot you made earlier. Try this out. To take care of the overlap, you might want to add some transparency in the geom_density() function using the alpha argument.

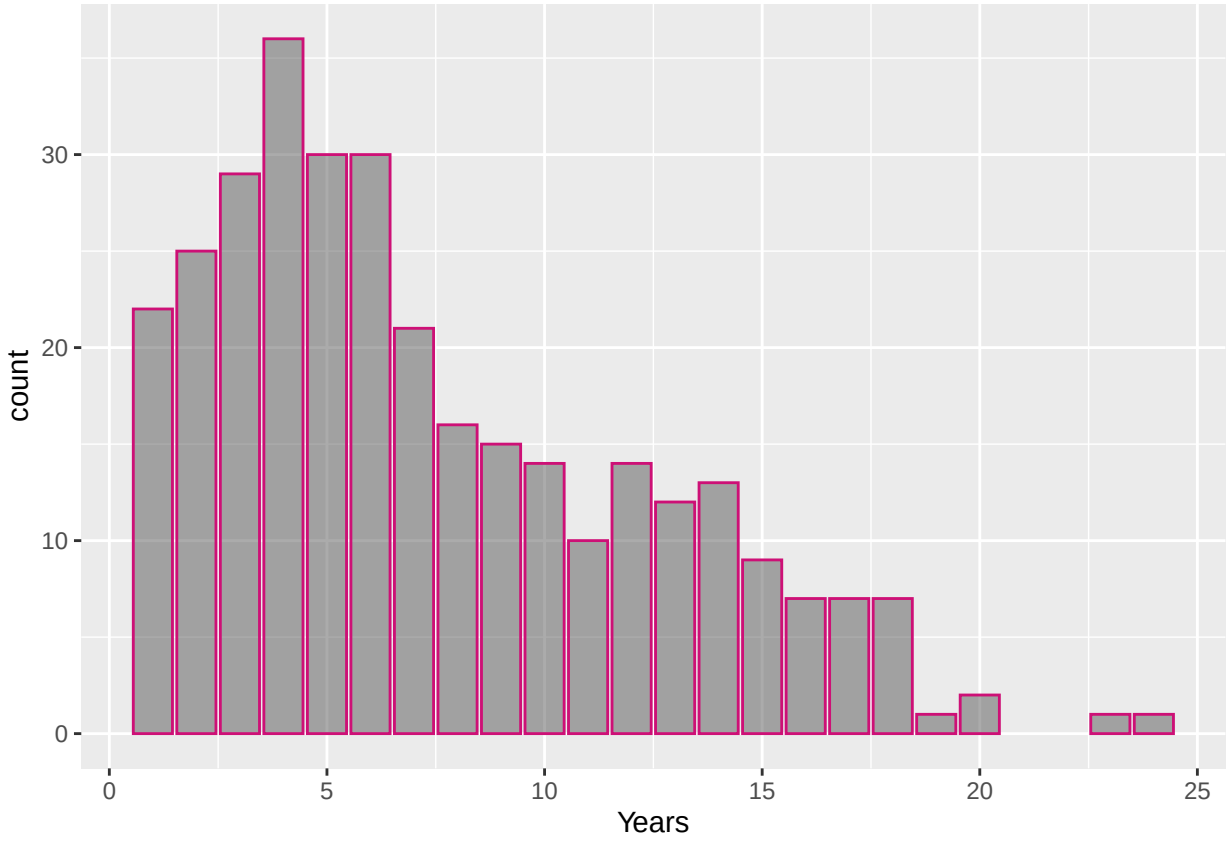
```
ggplot(gg_students, aes(grade, fill = programme, colour = programme)) +
geom_density(alpha = 0.2) +
  ylab(NULL) +
  xlim(3, 11) +
  theme_minimal(base_size = 11)
```



13. Create a bar plot of the variable Years from the Hitters dataset.

```
ggplot(Hitters) +
  geom_bar(mapping = aes(x = Years), binwidth = 0.75, color = "Deeppink3", alpha = 0.5)
```

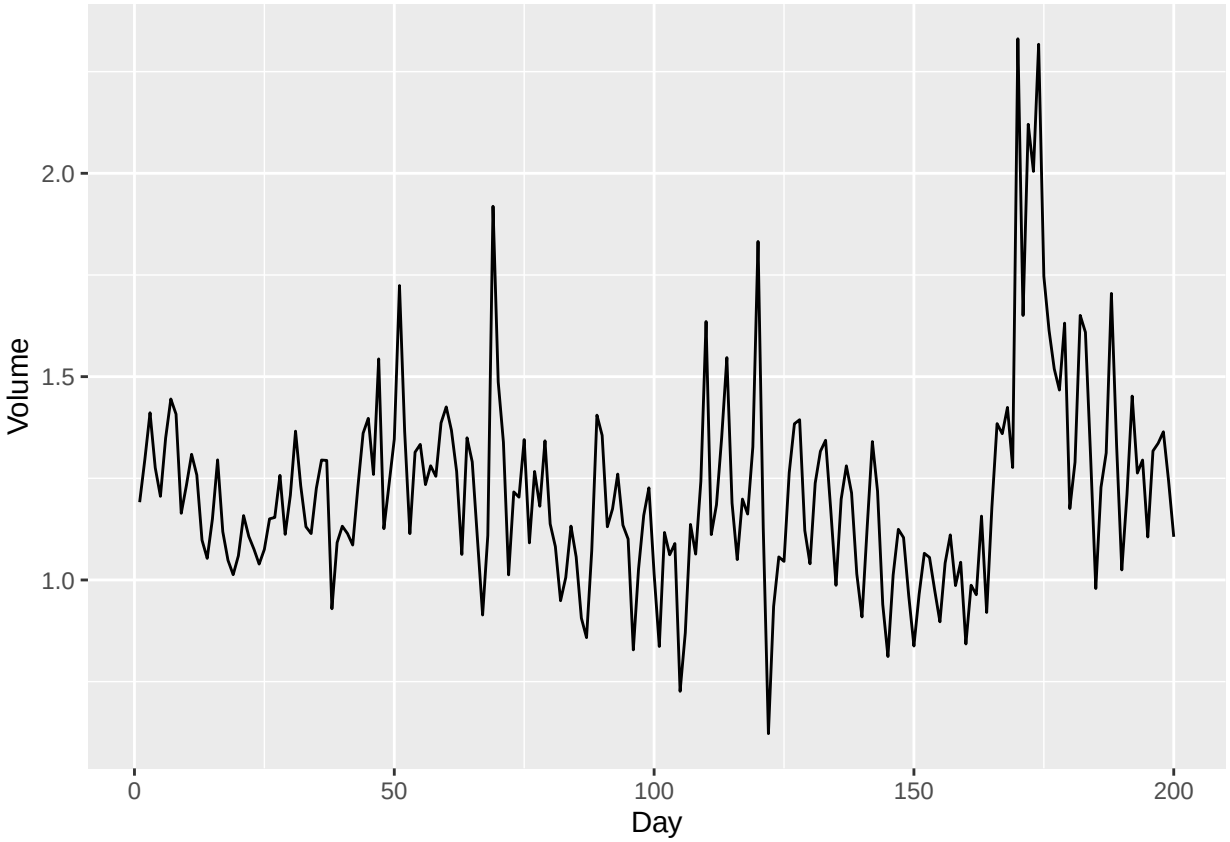
```
## Warning: Ignoring unknown parameters: binwidth
```



14. Use geom_line() to make a line plot out of the first 200 observations of the variable Volume (the number of trades made on each day) of the Smarket dataset. You will need to create a Day variable using mutate() to map to the x-position. This variable can simply be the integers from 1 to 200. Remember, you can select the first 200 rows using Smarket[1:200,].

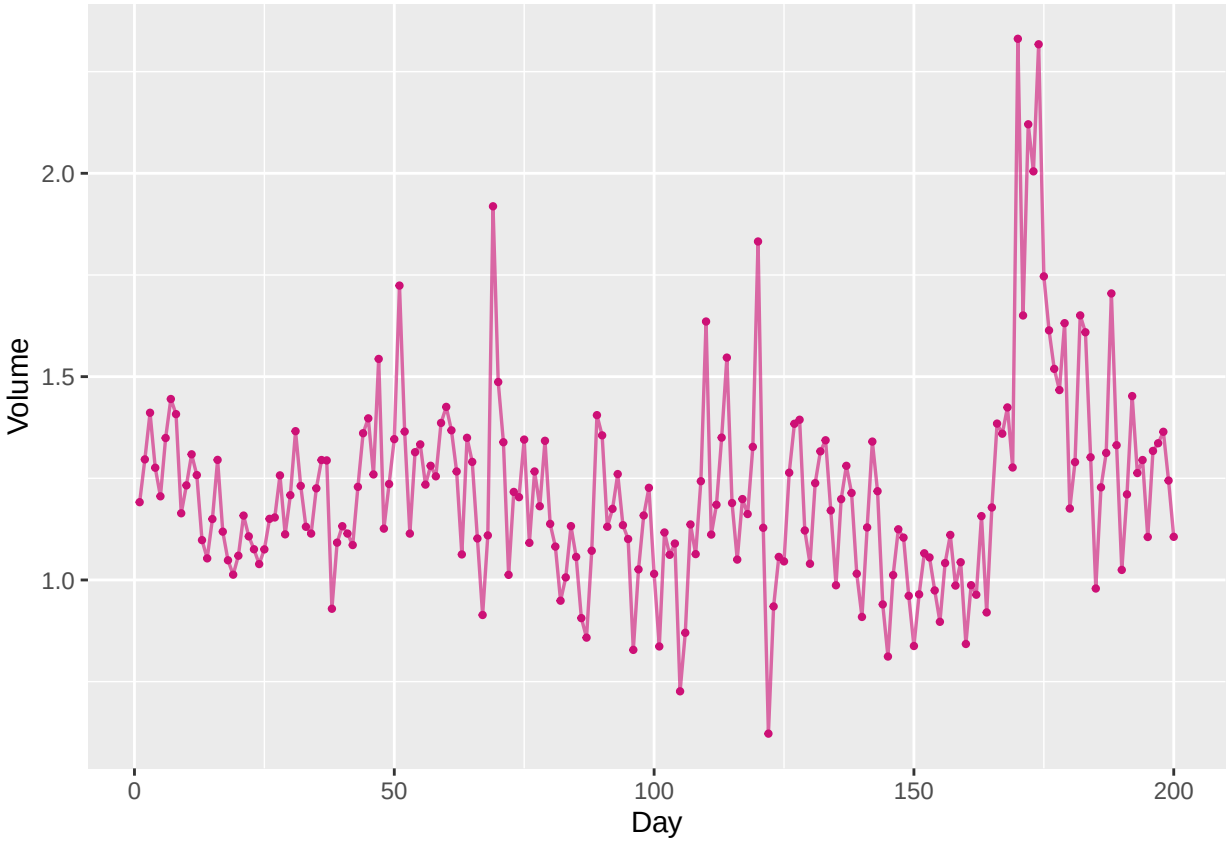
```
#Smarket$ID <- seq.int(nrow(Smarket))
Smarketshort <- Smarket[1:200, ] %>%
  mutate(Day = seq.int(nrow(Smarket[1:200, ])))

ggplot(data = Smarketshort) +
  geom_line( aes(x = Day, y = Volume))
```



15. Give the line a nice colour and increase its size. Also add points of the same colour on top.

```
ggplot(data = Smarketshort) +
  geom_line( aes(x = Day, y = Volume), colour = "deeppink3", size = 0.6, alpha = 0.6) +
  geom_point( aes(x = Day, y = Volume), colour = "deeppink3", size = 0.8)
```



16.Use the function which.max() to find out which of the first 200 days has the highest trade volume and use the function max() to find out how large this volume was.

```
#max day
which.max(Smarketshort$Volume)
```

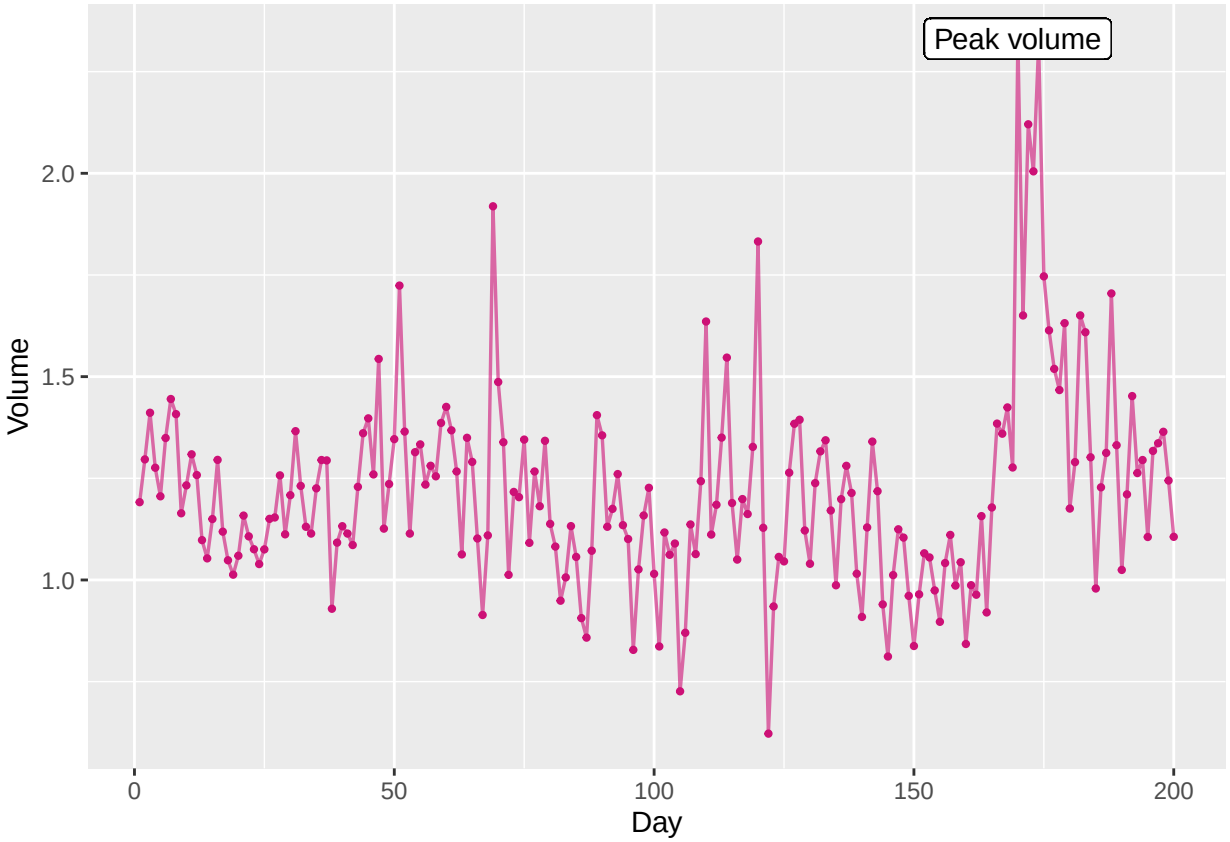
```
## [1] 170
```

```
#max vol
max(Smarketshort$Volume)
```

```
## [1] 2.33083
```

17. Use geom_label(aes(x = your_x, y = your_y, label = "Peak volume")) to add a label to this day. You can use either the values or call the functions. Place the label near the peak!

```
ggplot(data = Smarketshort) +
  geom_line( aes(x = Day, y = Volume), colour = "deeppink3", size = 0.6, alpha = 0.6) +
  geom_point( aes(x = Day, y = Volume), colour = "deeppink3", size = 0.8) +
  geom_label(aes(x = which.max(Volume), y = max(Volume), label = "Peak volume"))
```



18. Create a data frame called baseball based on the Hitters dataset. In this data frame, create a factor variable which splits players' salary range into 3 categories. Tip: use the filter() function to remove the missing values, and then use the cut() function and assign nice labels to the categories. In addition, create a variable which indicates the proportion of career hits that was a home run.

```
baseball <- Hitters %>% drop_na(Salary) %>%
  mutate(Wages=as.numeric(gsub("\\.", "", Salary))) %>%
  mutate(Cap=cut(Wages, breaks = c(0, 500000, 1000000, 5000000), labels = c("Amateur", '
  mutate(HRR=(HmRun/Runs))
```

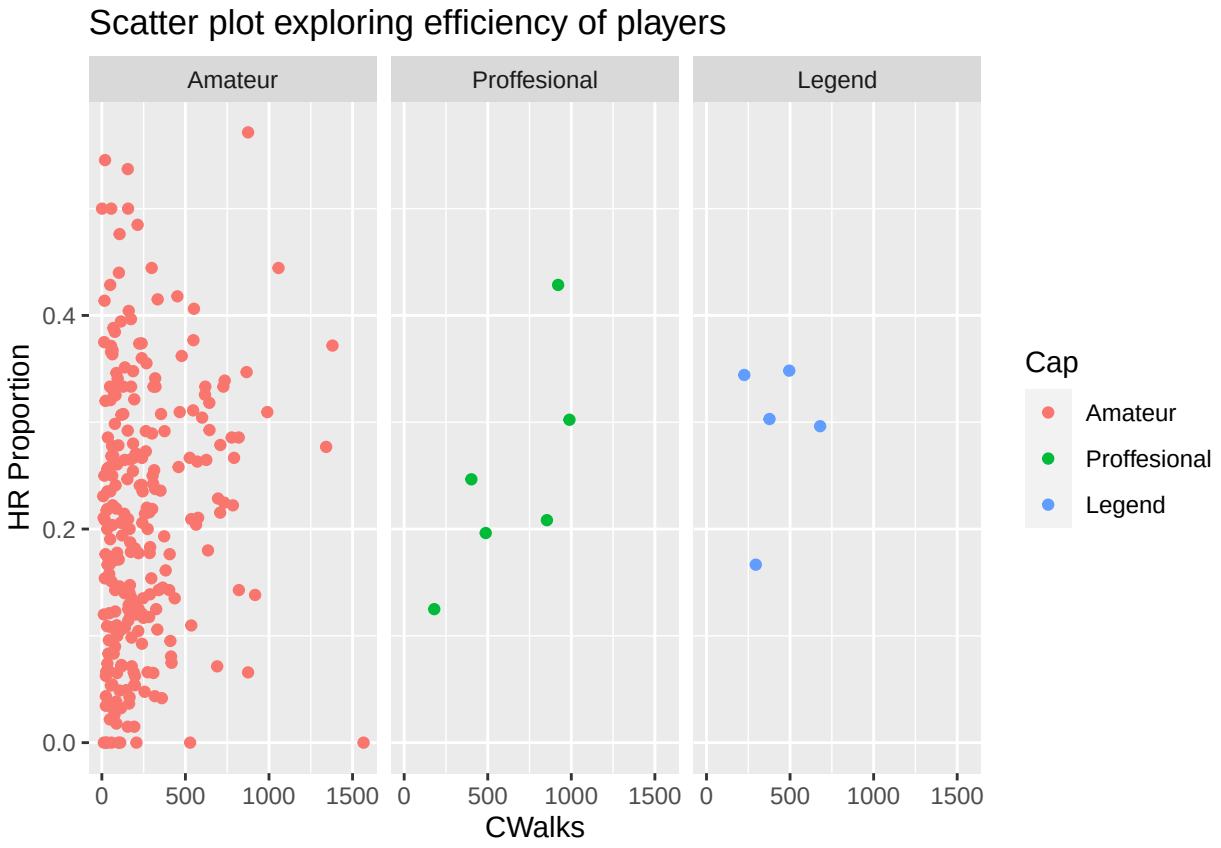
19. Create a scatter plot where you map CWalks to the x position and the proportion you calculated in the previous exercise to the y position. Fix the y axis limits to (0, 0.4) and the x axis to (0, 1600) using ylim() and xlim(). Add nice x and y axis titles using the labs() function. Save the plot as the variable baseball_plot.

```
baseball_plot <- ggplot(data = baseball,
       aes(x = CWalks, y = HRR),
       ylim(0,0.4),
       xlim(0, 1600),
       color = Cap) +
  geom_point(mapping = aes(color = Cap)) +
  labs(title = "Scatter plot exploring efficiency of players", x = "CWalks", y = "HR Pro
```

20. Split up this plot into three parts based on the salary range variable you calculated. Use the facet_wrap() function for this; look at the examples in the help file for tips.

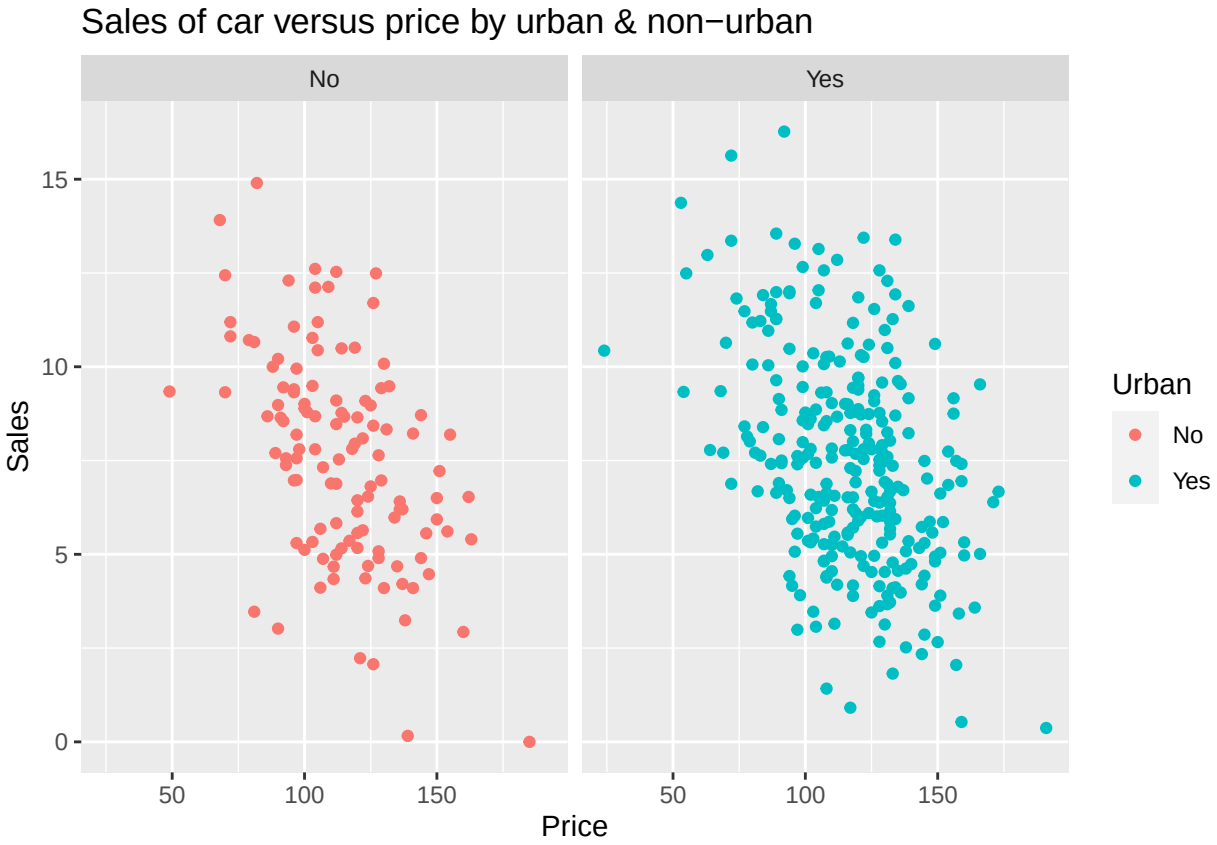
```
ggplot(data = baseball,
       aes(x = CWalks, y = HRR),
       ylim(0,0.4),
       xlim(0, 1600),
       color = Cap) +
  geom_point(mapping = aes(color = Cap)) +
  labs(title = "Scatter plot exploring efficiency of players", x = "CWalks", y = "HR Pro
  facet_wrap(~ Cap)
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



21. Create an interesting data visualisation based on the Carseats data from the ISLR package.

```
ggplot(Carseats, aes(x = Price, y = Sales), color = Urban) +
  geom_point(mapping = aes(color = Urban)) +
  labs(title = "Sales of car versus price by urban & non-urban") +
  facet_wrap(~ Urban)
```


Sales of car versus price by urban & non−urban
No
Yes
15
10
5
0
Sales
50
100
150
Price
Urban
No
Yes