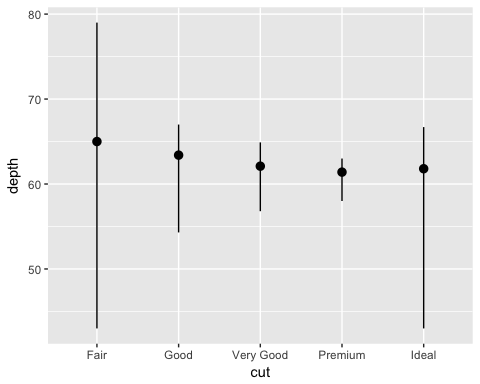
Homework 2

# 3.7.1 Exercises

## (a) Consider the following plot:

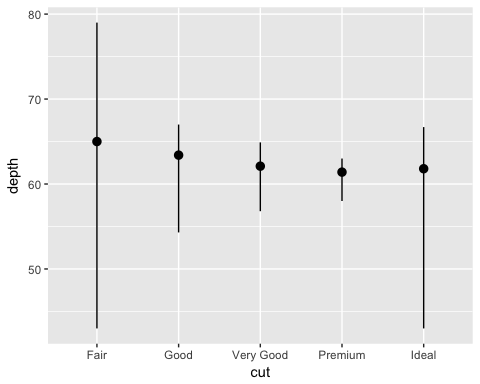
ggplot(data = diamonds) +  
 stat\_summary(  
 mapping = aes(x = cut, y = depth),  
 fun.ymin = min,  
 fun.ymax = max,  
 fun.y = median  
 )

 What is the default geom associated with stat\_summary()?

geom\_pointrange()

How could you rewrite the plot code so that it drew the same graph, but used the default geom instead of stat\_summary()?

ggplot(data = diamonds) +  
 geom\_pointrange(mapping = aes(x = cut, y = depth),  
 stat = "summary",  
 fun.ymin = min,  
 fun.ymax = max,  
 fun.y = median)



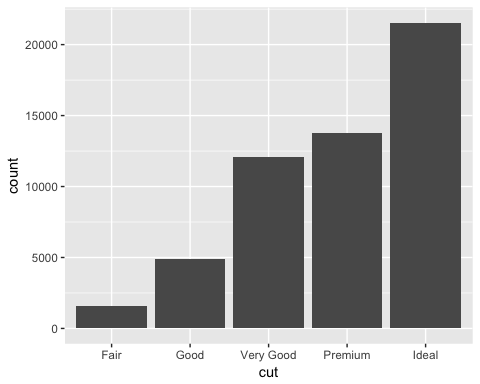
## (b) What does geom\_col() do, and how is it different from geom\_bar()?

geom\_col uses stat\_identity, so it will give you a bar graph with the heights of the bars representing the values of the data. geom\_bar, on the other hand, uses stat\_count to determine the heights of the bars, so it will give you bar heights proportional to how many cases there are within each group. If we were to set geom\_bar(stat = identity), this would be the same as geom\_col()

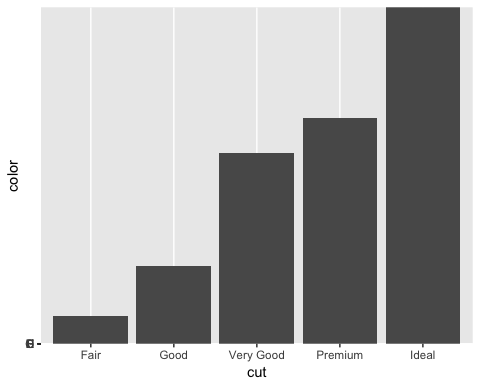
See: ?geom\_col or ?geom\_bar for more information.

Examples:

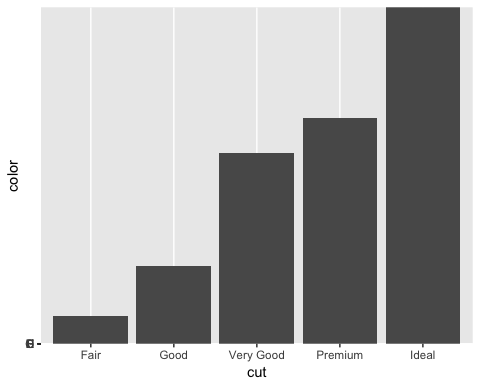
ggplot(diamonds, aes(cut)) + geom\_bar()



ggplot(diamonds, aes(x=cut, y=color)) + geom\_col()



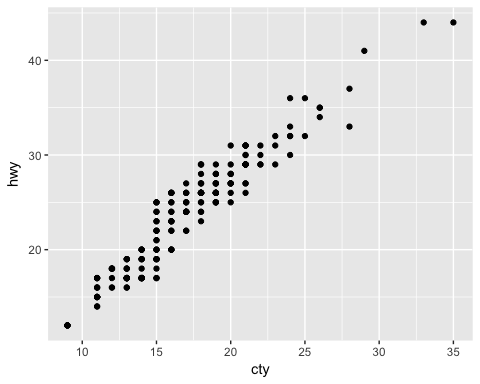
ggplot(diamonds, aes(x=cut, y=color)) + geom\_bar(stat = "identity")



# 3.8.1 Exercises

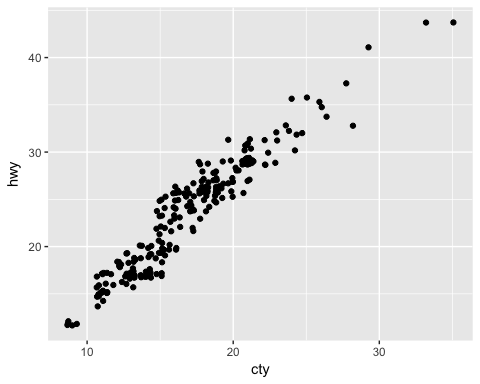
## (a) What is the problem with this plot? How could you improve it?

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +   
 geom\_point()



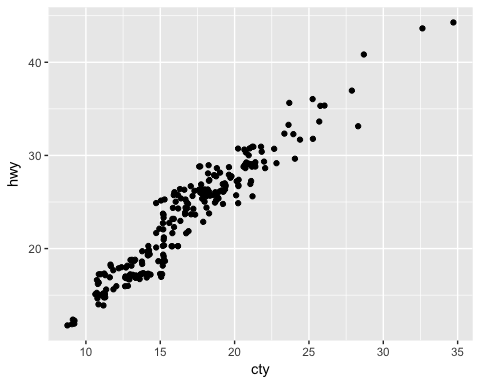
Many of the data points overlap because there are several that fall directly on top of one another. We can jitter or slightly scatter the points, which will improve the overall visualization.

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +   
 geom\_jitter()

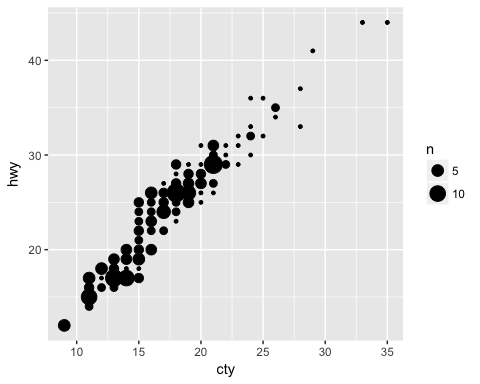


## (b) Compare and contrast geom\_jitter() with geom\_count().

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +   
 geom\_jitter()



ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +   
 geom\_count()

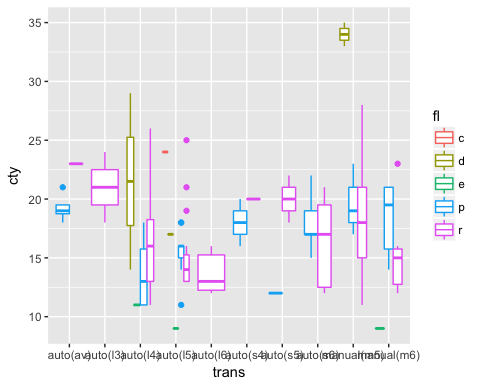


Rather than slightly scattering the points, geom\_count() counts the number of points at each location, then makes the point size proportional to how many points are at that location.

## (c) What is the default position adjustment for geom\_boxplot()? Create a visualization of the mpg dataset that demonstrates it.

The default position adjustment is position\_dodge().

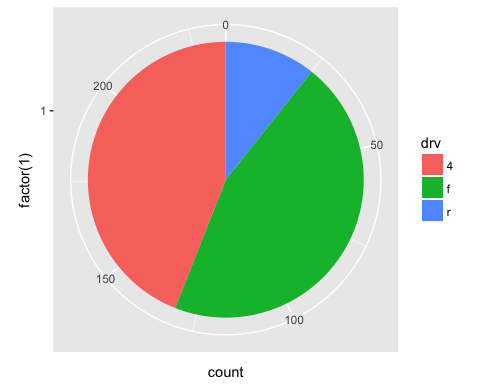
ggplot(data = mpg, mapping = aes(x = trans, y = cty, color = fl)) +   
 geom\_boxplot(position = "dodge")



# 3.9.1 Exercises

## (a) Turn a stacked bar chart into a pie chart using coord\_polar().

ggplot(data = mpg, mapping = aes(x=factor(1), fill = drv)) +  
 geom\_bar(width = 1) +  
 coord\_polar(theta = "y")

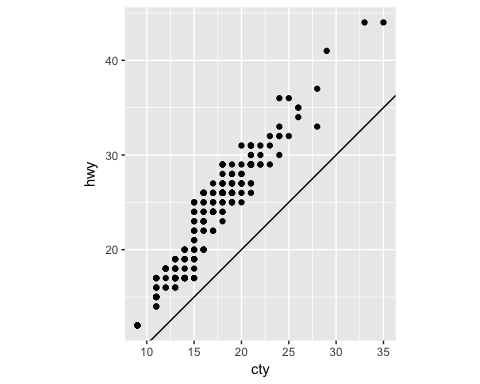


## (b) What is the difference between coord\_quickmap() and coord\_map()? ----

coord\_map() projects a portion of the 3D earth onto a flat plane. coord\_map() has to project all of the elements of the graphic. coord\_quickmap() preserves lines of latitude and longitude as straight and is therefore faster to draw, but it is also a little less accurate.

## (c) What does the plot below tell you about the relationship between city and highway mpg? Why is coord\_fixed() important? What does geom\_abline() do?

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +  
 geom\_point() +   
 geom\_abline() +  
 coord\_fixed()



The relationship seems to be linear with cars having slightly better highway mileage than city mileage. coord\_fixed() makes the plat have equal intervals on the x and y axes so they are directly comparable. geom\_abline() draws a line that has a slope of 1. This helps us visualize automobile gas efficiency and conclude that on average, highway is slightly higher than city.

# 5.2.4 Exercises

## (a) For the nycflights13::flights dataset, find all flights that:

### (1) Had an arrival delay of two or more hours

library(nycflights13)  
filter(flights, arr\_delay>=120)

## # A tibble: 10,200 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 811 630 101 1047  
## 2 2013 1 1 848 1835 853 1001  
## 3 2013 1 1 957 733 144 1056  
## 4 2013 1 1 1114 900 134 1447  
## 5 2013 1 1 1505 1310 115 1638  
## 6 2013 1 1 1525 1340 105 1831  
## 7 2013 1 1 1549 1445 64 1912  
## 8 2013 1 1 1558 1359 119 1718  
## 9 2013 1 1 1732 1630 62 2028  
## 10 2013 1 1 1803 1620 103 2008  
## # ... with 10,190 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (2) Flew to Houston (IAH or HOU)

filter(flights, dest=="IAH" | dest=="HOU")

## # A tibble: 9,313 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 623 627 -4 933  
## 4 2013 1 1 728 732 -4 1041  
## 5 2013 1 1 739 739 0 1104  
## 6 2013 1 1 908 908 0 1228  
## 7 2013 1 1 1028 1026 2 1350  
## 8 2013 1 1 1044 1045 -1 1352  
## 9 2013 1 1 1114 900 134 1447  
## 10 2013 1 1 1205 1200 5 1503  
## # ... with 9,303 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (3) Were operated by United, American, or Delta

filter(flights, carrier=="UA" |  
 carrier=="AA" |  
 carrier=="DL")

## # A tibble: 139,504 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 554 600 -6 812  
## 5 2013 1 1 554 558 -4 740  
## 6 2013 1 1 558 600 -2 753  
## 7 2013 1 1 558 600 -2 924  
## 8 2013 1 1 558 600 -2 923  
## 9 2013 1 1 559 600 -1 941  
## 10 2013 1 1 559 600 -1 854  
## # ... with 139,494 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (4) Departed in summer (July, August, and September)

filter(flights, month>=7, month<=9)

## # A tibble: 86,326 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 7 1 1 2029 212 236  
## 2 2013 7 1 2 2359 3 344  
## 3 2013 7 1 29 2245 104 151  
## 4 2013 7 1 43 2130 193 322  
## 5 2013 7 1 44 2150 174 300  
## 6 2013 7 1 46 2051 235 304  
## 7 2013 7 1 48 2001 287 308  
## 8 2013 7 1 58 2155 183 335  
## 9 2013 7 1 100 2146 194 327  
## 10 2013 7 1 100 2245 135 337  
## # ... with 86,316 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (5) Arrived more than two hours late, but didn’t leave late

filter(flights, arr\_delay>=120, dep\_delay<=0)

## # A tibble: 29 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 27 1419 1420 -1 1754  
## 2 2013 10 7 1350 1350 0 1736  
## 3 2013 10 7 1357 1359 -2 1858  
## 4 2013 10 16 657 700 -3 1258  
## 5 2013 11 1 658 700 -2 1329  
## 6 2013 3 18 1844 1847 -3 39  
## 7 2013 4 17 1635 1640 -5 2049  
## 8 2013 4 18 558 600 -2 1149  
## 9 2013 4 18 655 700 -5 1213  
## 10 2013 5 22 1827 1830 -3 2217  
## # ... with 19 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (6) Were delayed by at least an hour, but made up over 30 minutes in flight

filter(flights, dep\_delay>=60, dep\_delay-arr\_delay>=30)

## # A tibble: 2,074 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 1716 1545 91 2140  
## 2 2013 1 1 2205 1720 285 46  
## 3 2013 1 1 2326 2130 116 131  
## 4 2013 1 3 1503 1221 162 1803  
## 5 2013 1 3 1821 1530 171 2131  
## 6 2013 1 3 1839 1700 99 2056  
## 7 2013 1 3 1850 1745 65 2148  
## 8 2013 1 3 1923 1815 68 2036  
## 9 2013 1 3 1941 1759 102 2246  
## 10 2013 1 3 1950 1845 65 2228  
## # ... with 2,064 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

### (7) Departed between midnight and 6am (inclusive)

filter(flights, dep\_time>=0, dep\_time<=600)

## # A tibble: 9,344 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 9,334 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

## (b) Why is NA ^ 0 not missing? Why is NA | TRUE not missing? Why is FALSE & NA not missing? Can you figure out the general rule? (NA \* 0 is a tricky counterexample!)

*NA ^ 0 - anything to the 0th power is 1.* NA | TRUE - as long as one condition is TRUE, the result is TRUE. TRUE is TRUE, thus this is always TRUE. \*FALSE & NA - NA indicates the absence of a value, so it is ignored.

Any operation on a missing value (NA) becomes a missing value. For example, NA \* 0 is NA. In conditional expressions, missing values are ignored.

# 5.3.1 Exercises

## (a) How could you use arrange() to sort all missing values to the start? (Hint: use is.na()).

arrange(flights, !is.na(dep\_time))

## # A tibble: 336,776 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 NA 1630 NA NA  
## 2 2013 1 1 NA 1935 NA NA  
## 3 2013 1 1 NA 1500 NA NA  
## 4 2013 1 1 NA 600 NA NA  
## 5 2013 1 2 NA 1540 NA NA  
## 6 2013 1 2 NA 1620 NA NA  
## 7 2013 1 2 NA 1355 NA NA  
## 8 2013 1 2 NA 1420 NA NA  
## 9 2013 1 2 NA 1321 NA NA  
## 10 2013 1 2 NA 1545 NA NA  
## # ... with 336,766 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

## (b) Which flights travelled the longest distance? Which travelled the shortest?

arrange(flights, desc(distance))

## # A tibble: 336,776 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 857 900 -3 1516  
## 2 2013 1 2 909 900 9 1525  
## 3 2013 1 3 914 900 14 1504  
## 4 2013 1 4 900 900 0 1516  
## 5 2013 1 5 858 900 -2 1519  
## 6 2013 1 6 1019 900 79 1558  
## 7 2013 1 7 1042 900 102 1620  
## 8 2013 1 8 901 900 1 1504  
## 9 2013 1 9 641 900 1301 1242  
## 10 2013 1 10 859 900 -1 1449  
## # ... with 336,766 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

arrange(flights, distance)

## # A tibble: 336,776 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 7 27 NA 106 NA NA  
## 2 2013 1 3 2127 2129 -2 2222  
## 3 2013 1 4 1240 1200 40 1333  
## 4 2013 1 4 1829 1615 134 1937  
## 5 2013 1 4 2128 2129 -1 2218  
## 6 2013 1 5 1155 1200 -5 1241  
## 7 2013 1 6 2125 2129 -4 2224  
## 8 2013 1 7 2124 2129 -5 2212  
## 9 2013 1 8 2127 2130 -3 2304  
## 10 2013 1 9 2126 2129 -3 2217  
## # ... with 336,766 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

# 5.4.1 Exercises

## (a) Brainstorm as many ways as possible to select dep\_time, dep\_delay, arr\_time, and arr\_delay from flights.

select(flights, dep\_time, dep\_delay, arr\_time, arr\_delay)

## # A tibble: 336,776 × 4  
## dep\_time dep\_delay arr\_time arr\_delay  
## <int> <dbl> <int> <dbl>  
## 1 517 2 830 11  
## 2 533 4 850 20  
## 3 542 2 923 33  
## 4 544 -1 1004 -18  
## 5 554 -6 812 -25  
## 6 554 -4 740 12  
## 7 555 -5 913 19  
## 8 557 -3 709 -14  
## 9 557 -3 838 -8  
## 10 558 -2 753 8  
## # ... with 336,766 more rows

select(flights, ends\_with("delay"))

## # A tibble: 336,776 × 2  
## dep\_delay arr\_delay  
## <dbl> <dbl>  
## 1 2 11  
## 2 4 20  
## 3 2 33  
## 4 -1 -18  
## 5 -6 -25  
## 6 -4 12  
## 7 -5 19  
## 8 -3 -14  
## 9 -3 -8  
## 10 -2 8  
## # ... with 336,766 more rows

select(flights, contains("delay"))

## # A tibble: 336,776 × 2  
## dep\_delay arr\_delay  
## <dbl> <dbl>  
## 1 2 11  
## 2 4 20  
## 3 2 33  
## 4 -1 -18  
## 5 -6 -25  
## 6 -4 12  
## 7 -5 19  
## 8 -3 -14  
## 9 -3 -8  
## 10 -2 8  
## # ... with 336,766 more rows

select(flights, starts\_with("dep"), starts\_with("arr"))

## # A tibble: 336,776 × 4  
## dep\_time dep\_delay arr\_time arr\_delay  
## <int> <dbl> <int> <dbl>  
## 1 517 2 830 11  
## 2 533 4 850 20  
## 3 542 2 923 33  
## 4 544 -1 1004 -18  
## 5 554 -6 812 -25  
## 6 554 -4 740 12  
## 7 555 -5 913 19  
## 8 557 -3 709 -14  
## 9 557 -3 838 -8  
## 10 558 -2 753 8  
## # ... with 336,766 more rows

## (b) What does the one\_of() function do? Why might it be helpful in conjunction with this vector?

vars <- c("year", "month", "day", "dep\_delay", "arr\_delay")

It selects any variable that matches with any item in the vector.

select(flights, one\_of(vars))

## # A tibble: 336,776 × 5  
## year month day dep\_delay arr\_delay  
## <int> <int> <int> <dbl> <dbl>  
## 1 2013 1 1 2 11  
## 2 2013 1 1 4 20  
## 3 2013 1 1 2 33  
## 4 2013 1 1 -1 -18  
## 5 2013 1 1 -6 -25  
## 6 2013 1 1 -4 12  
## 7 2013 1 1 -5 19  
## 8 2013 1 1 -3 -14  
## 9 2013 1 1 -3 -8  
## 10 2013 1 1 -2 8  
## # ... with 336,766 more rows

# 5.5.2 Exercises

## (a) Currently dep\_time and sched\_dep\_time are convenient to look at, but hard to compute with because they’re not really continuous numbers. Convert them to a more convenient representation of number of minutes since midnight.

transmute(flights,  
 sched\_dep\_time = (sched\_dep\_time %/% 100)\*60 + sched\_dep\_time %% 100,  
 dep\_time = (dep\_time %/% 100)\*60 + dep\_time %% 100)

## # A tibble: 336,776 × 2  
## sched\_dep\_time dep\_time  
## <dbl> <dbl>  
## 1 315 317  
## 2 329 333  
## 3 340 342  
## 4 345 344  
## 5 360 354  
## 6 358 354  
## 7 360 355  
## 8 360 357  
## 9 360 357  
## 10 360 358  
## # ... with 336,766 more rows

## (b) Compare air\_time with arr\_time - dep\_time. What do you expect to see? What do you see? What do you need to do to fix it?

flights2 <- select(flights, air\_time, arr\_time, dep\_time)  
mutate(flights2, air\_time\_new = arr\_time-dep\_time)

## # A tibble: 336,776 × 4  
## air\_time arr\_time dep\_time air\_time\_new  
## <dbl> <int> <int> <int>  
## 1 227 830 517 313  
## 2 227 850 533 317  
## 3 160 923 542 381  
## 4 183 1004 544 460  
## 5 116 812 554 258  
## 6 150 740 554 186  
## 7 158 913 555 358  
## 8 53 709 557 152  
## 9 140 838 557 281  
## 10 138 753 558 195  
## # ... with 336,766 more rows

dep\_time and arr\_time are not measured in minutes. Therefore, these are not the same. We must make these variable continuous to calculate air\_time.