

COMPSCIX 415.2 Homework 3

Santosh Kanutala

6/22/2018

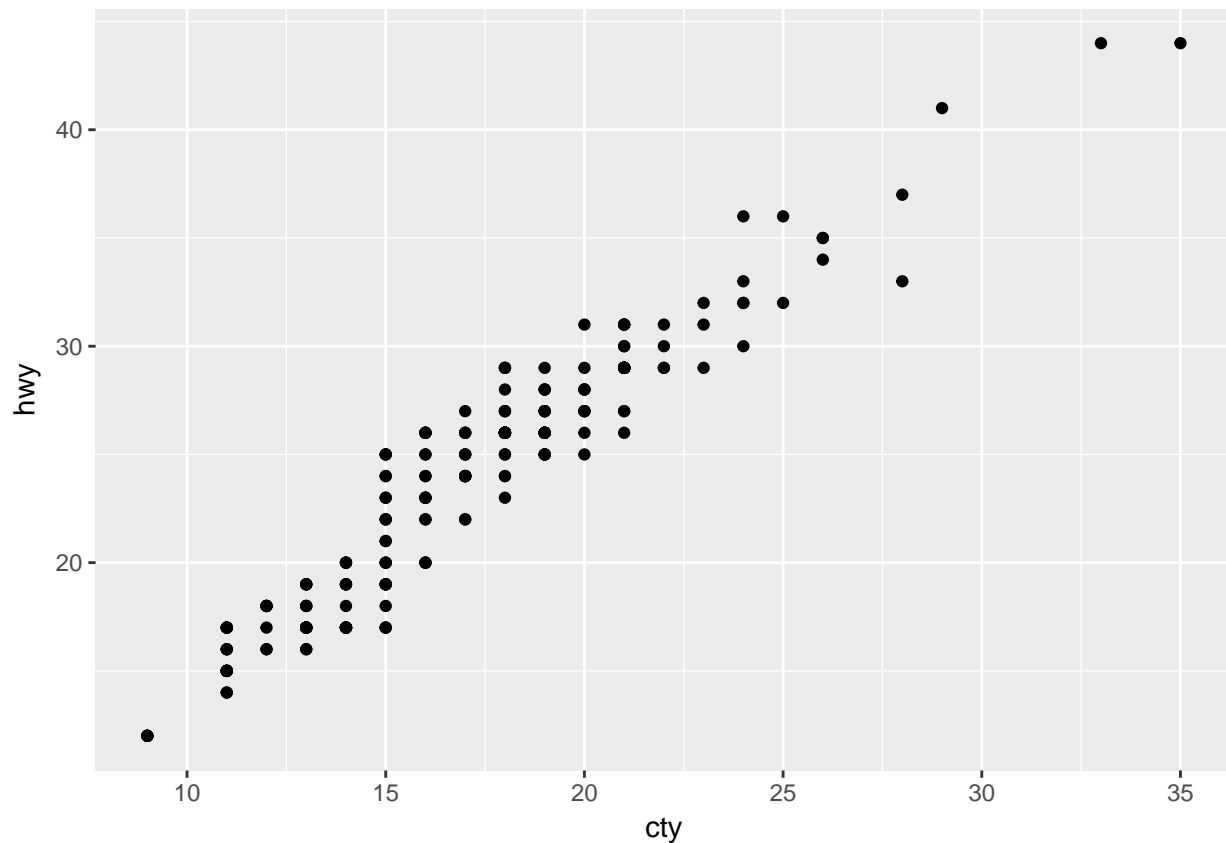
****My Github repository for my assignments can be found at below URL: (<https://github.com/santumagic/compscix-415-2assignments.git>)****

```
library(tidyverse)
library(mdsr)
```

Section 3.8.1: all exercises

QUESTION 1:

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +  
  geom_point()
```



ANSWER:

From the mpg dataset we know that cty and hwy both are continuous variables and when we plot them in a single plot, many data points will be overlapped especially for larger datasets. We can resolve this issue

(overplotting) by using adjustment to jitter with position = “jitter” or by using `geom_jitter()` as shown below.

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) + geom_point() + geom_jitter()
```

QUESTION 2:

ANSWER:

Lets find from the help function.

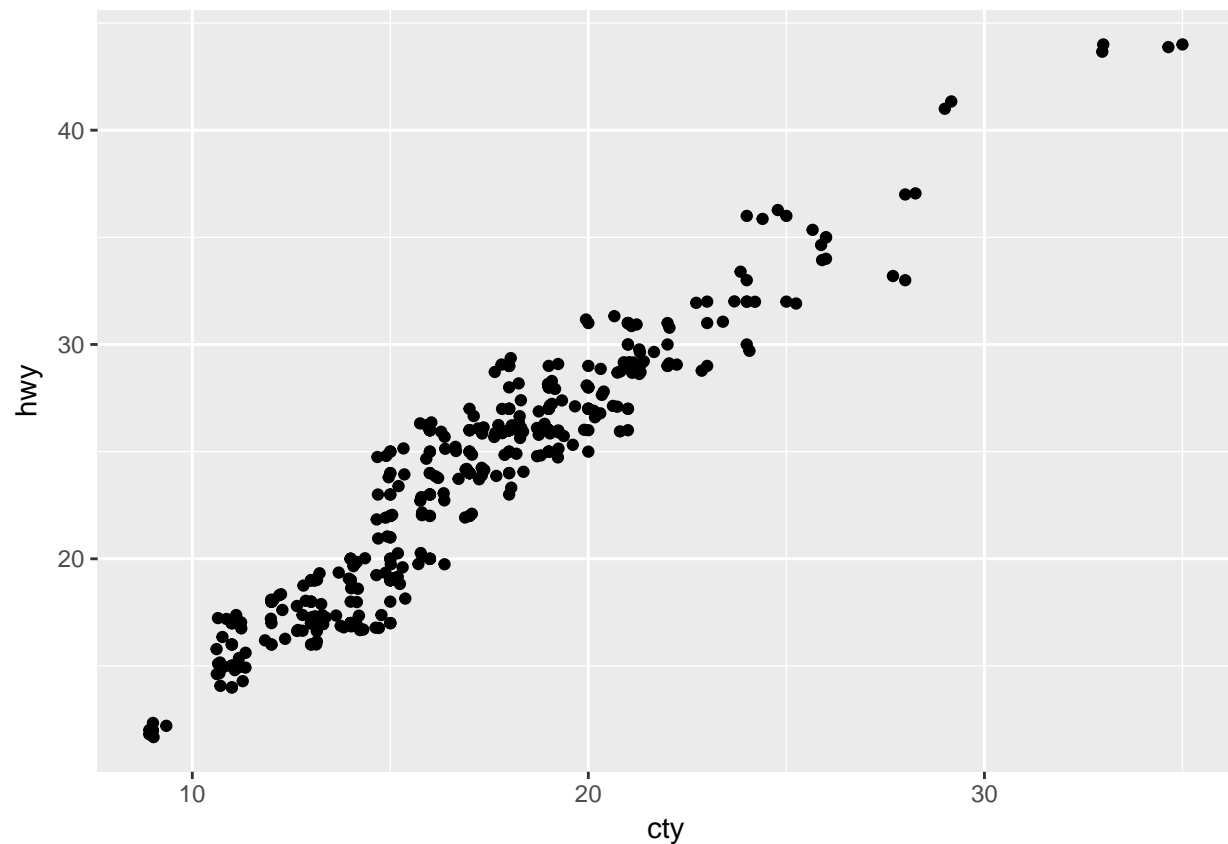
```
?geom_jitter
```

width and height are the parameters that control the jittering.

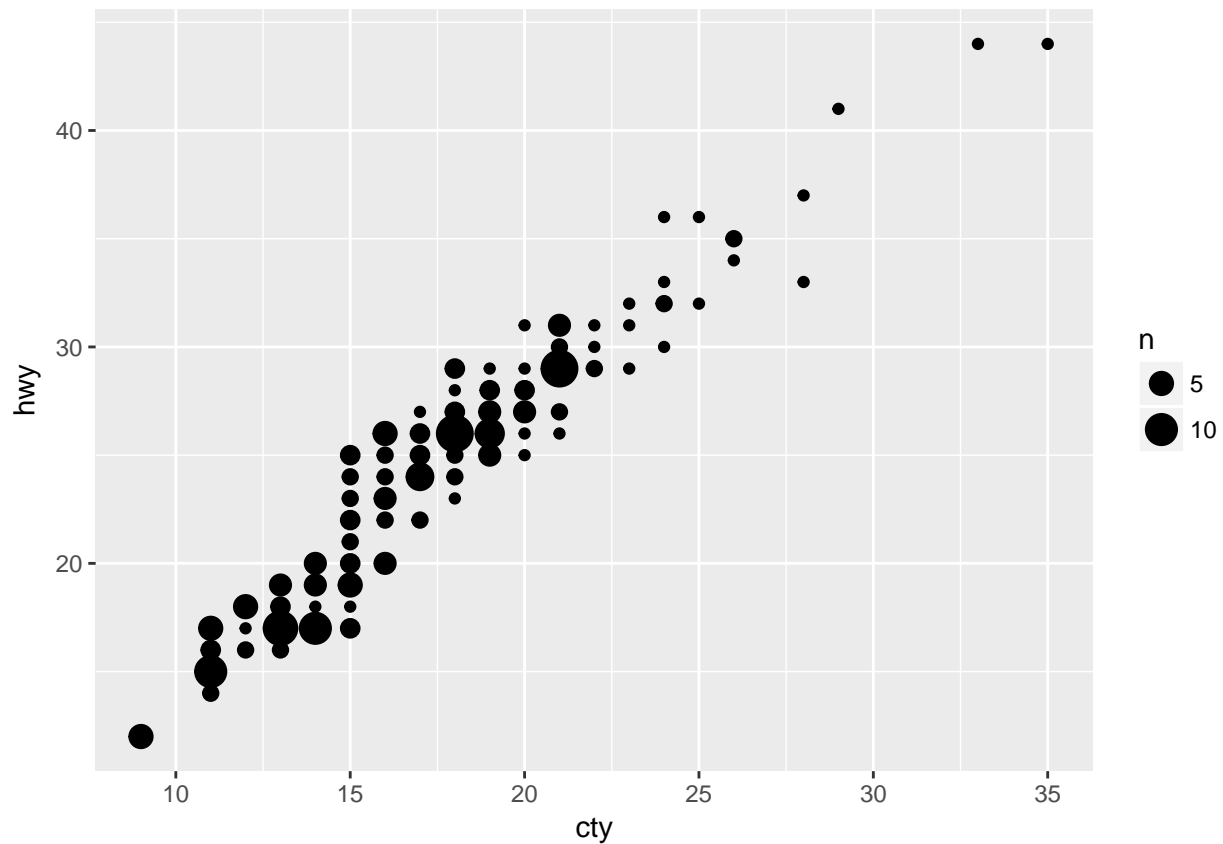
QUESTION 3:

ANSWER:

```
# geom_jitter()
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_point() +
  geom_jitter()
```



```
# geom_count()
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_point() +
  geom_count()
```



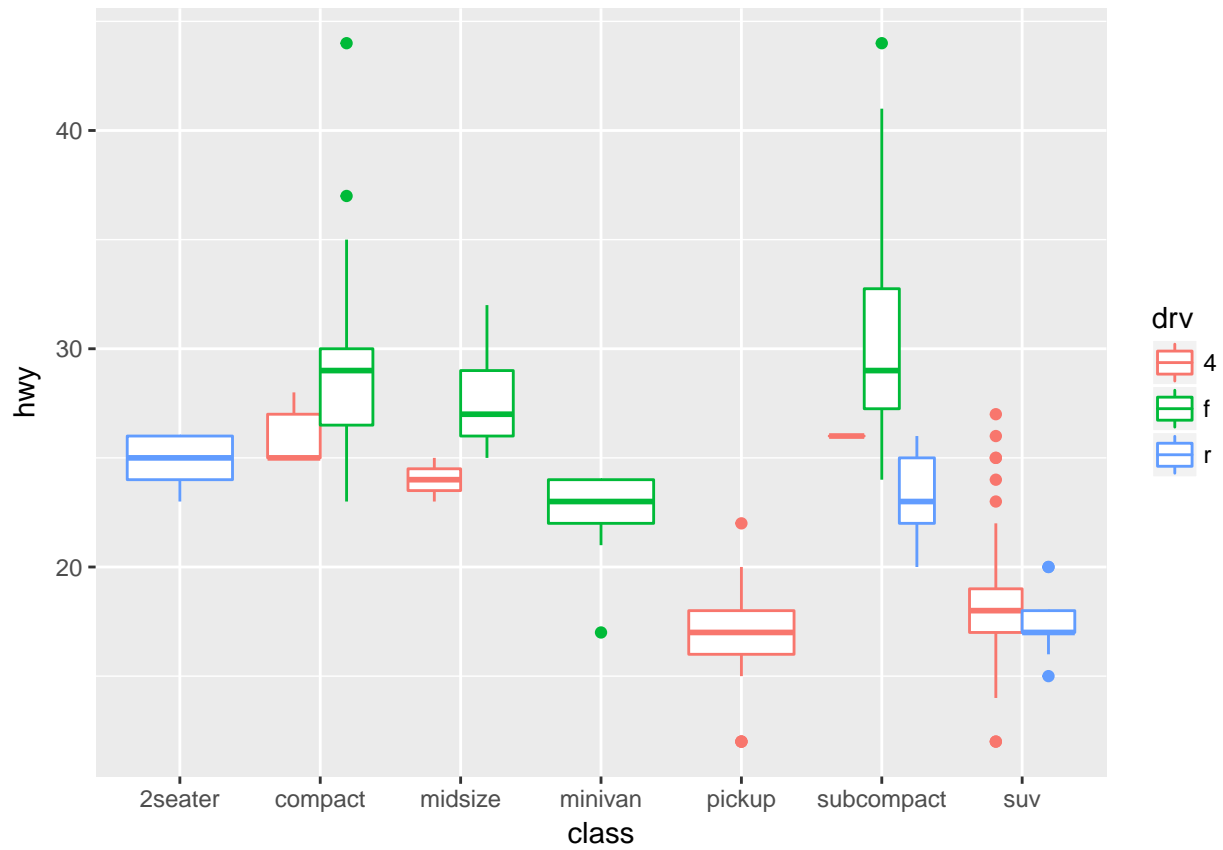
`geom_count()` is variant of `geom_point` and it basically it counts the number of data elements or observations at a point in the plot and then maps that count to the pointing area.

QUESTION 4:

ANSWER:

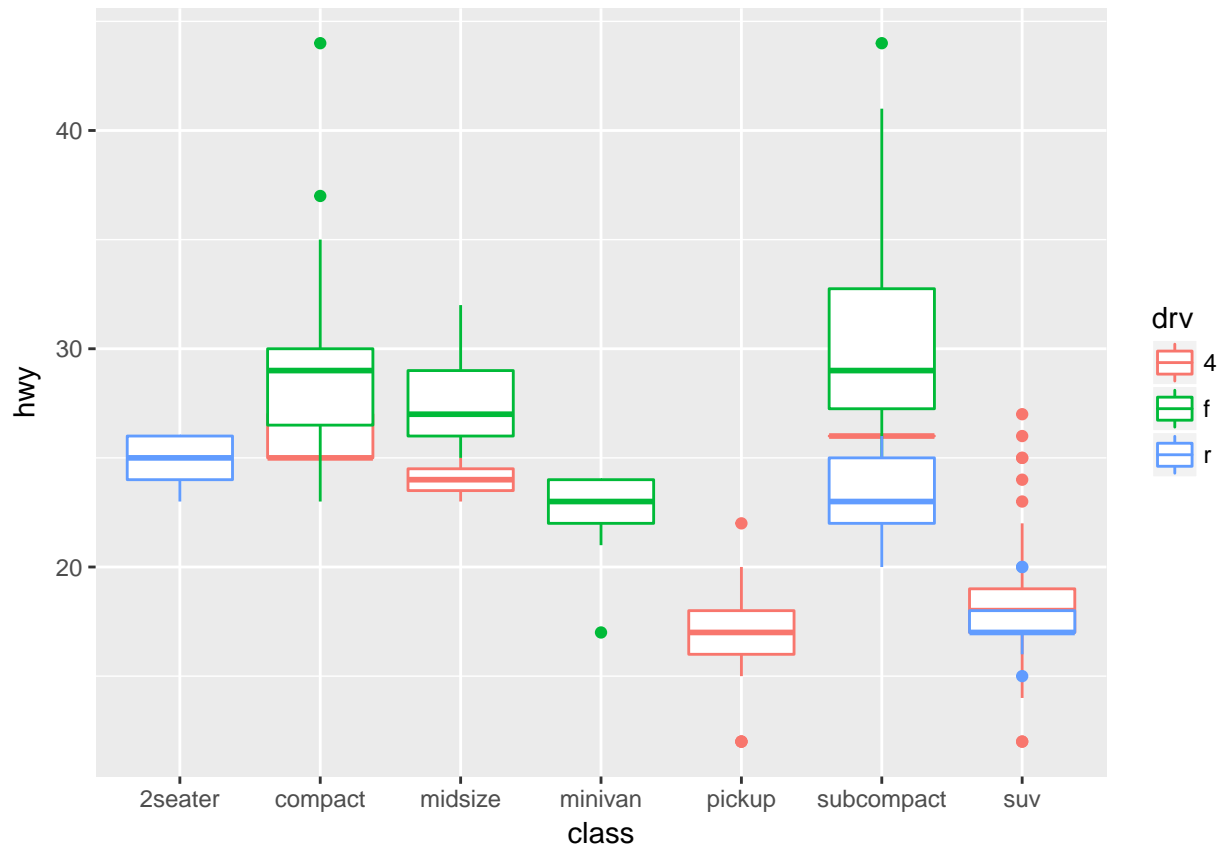
By observing all the below graphs, we can conclude that **position = “dodge”** is the default position adjustment for a boxplot.

```
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot()
```

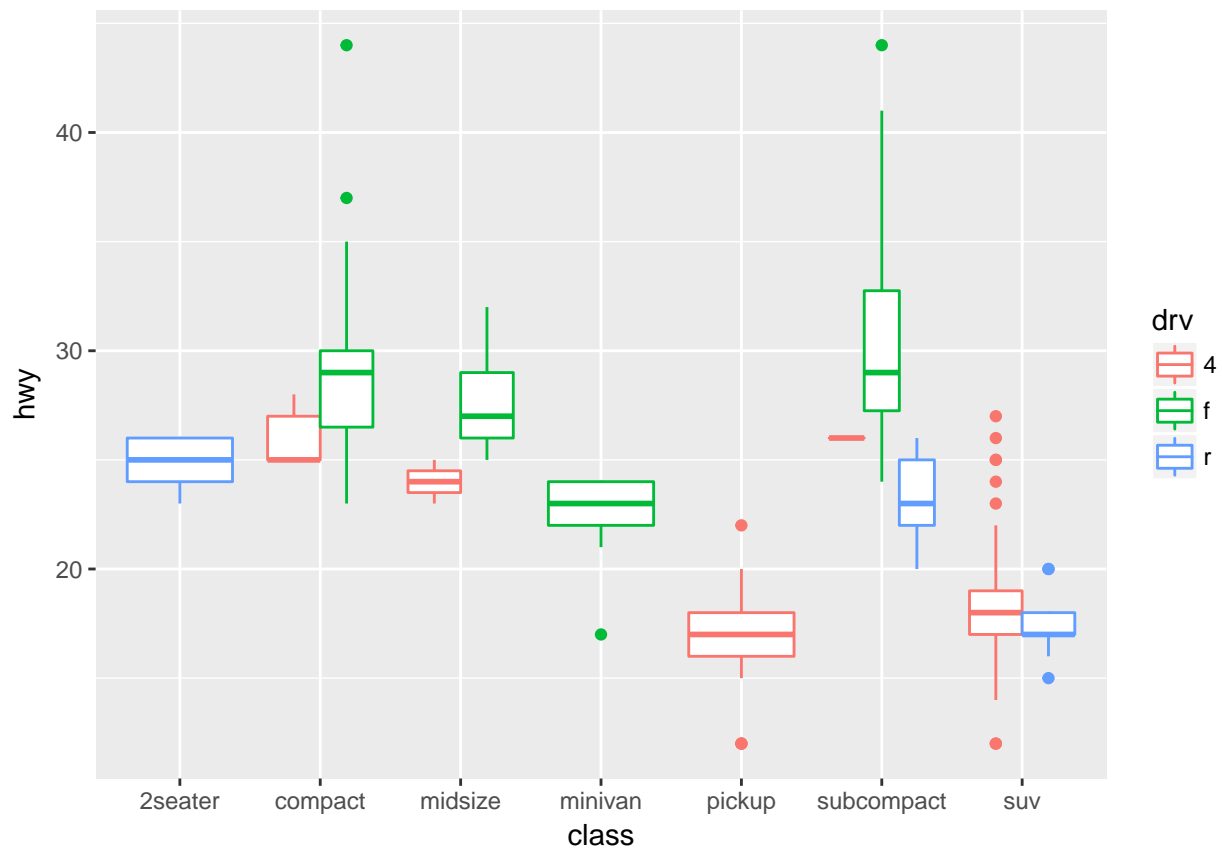


```
# lets try all types of position adjustments

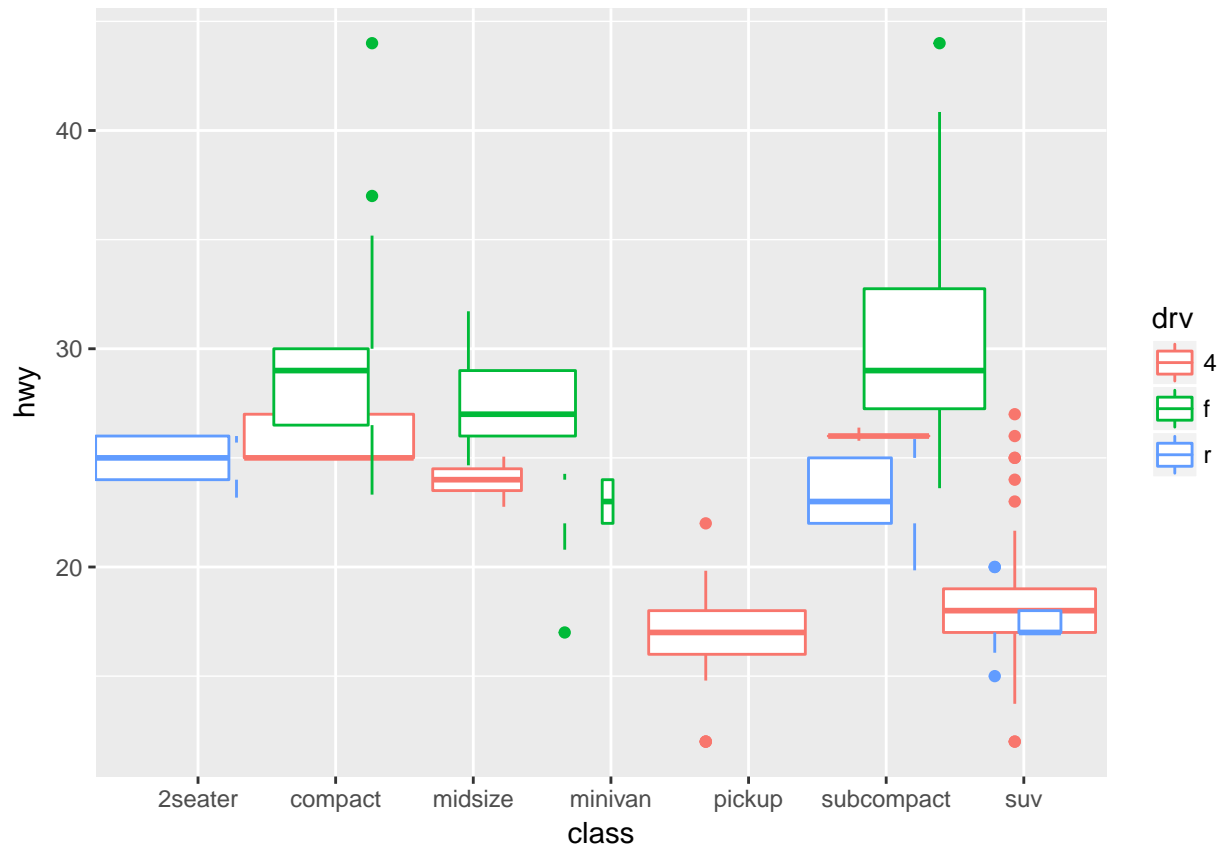
# position = "Identity"
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot(position = "Identity")
```



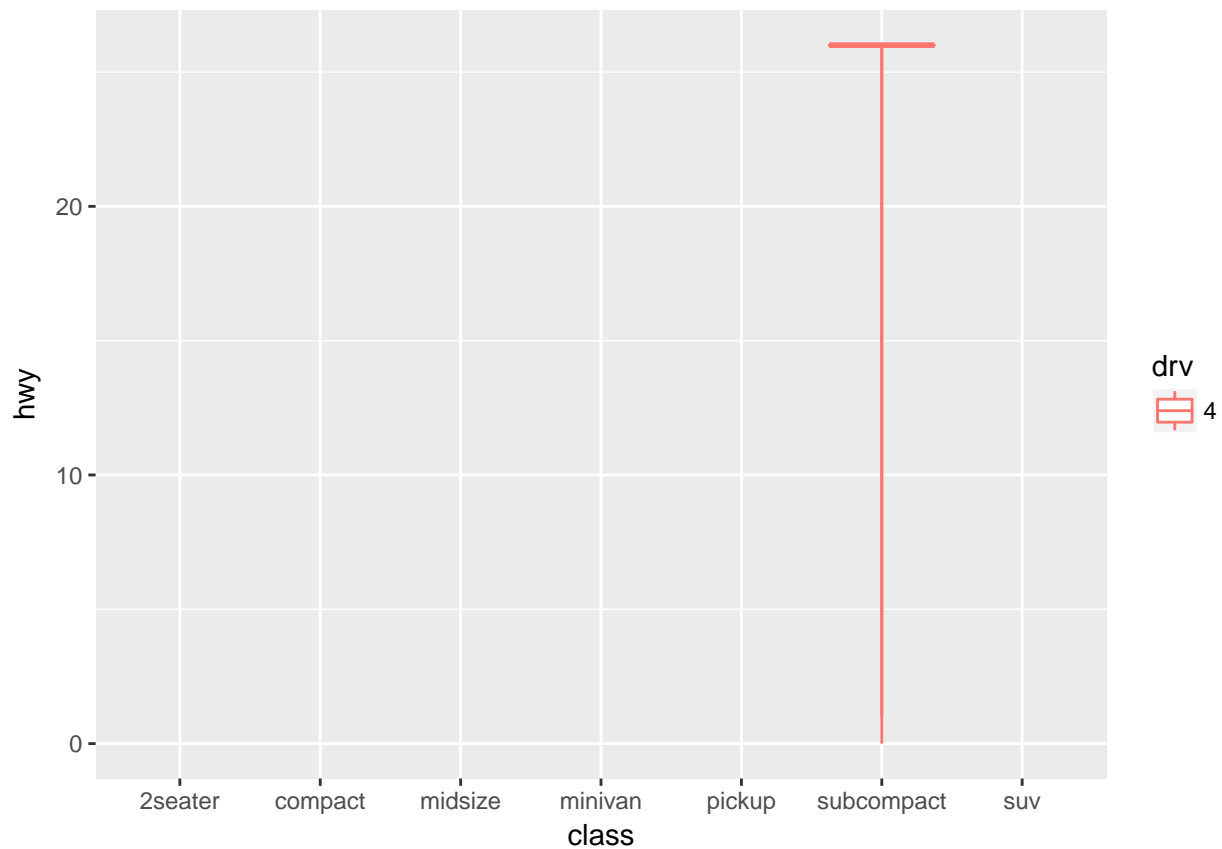
```
# position = "dodge"
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot(position = "dodge")
```



```
# position = "jitter"
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot(position = "jitter")
```



```
# position = "fill"
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot(position = "fill")
```



Section 3.9.1: #2 and #4 only

QUESTION 2:

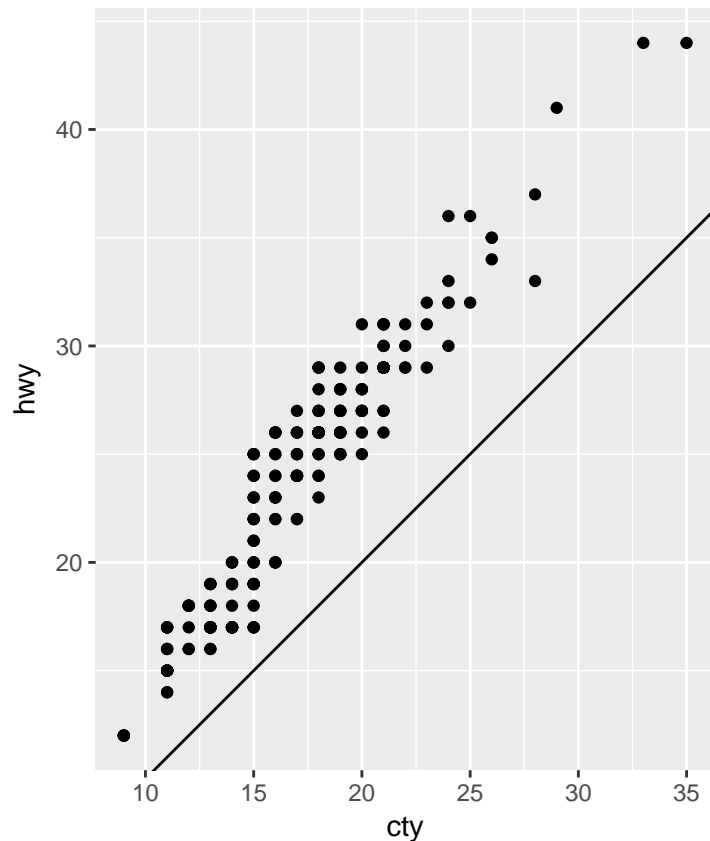
```
?labs ()
```

ANSWER:

labs will change the labels for the axes. In addition we can use this for titles and subtitles as well.

QUESTION 4:

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_point() +
  geom_abline() +
  coord_fixed()
```

ANSWER:

- From the above graph it is observed that the both the variables are positively related to each other.
- `coord_fixed` is important because it is making sure that the coordinates for the both variables are fixed.
- `geom_abline` plots the slope between the variables `cty` and `hwy`.

Section 4.4: #1 and #2 only

QUESTION 1:

```
my_variable <- 10 my_var1able
```

ANSWER:

There is a type in the second line. It should be : `my_variable`

QUESTION 2:

ANSWER:

```
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy))
filter(mpg, cyl == 8)
filter(diamonds, carat > 3)
```

Data transformation

```
library(nycflights13)
library(tidyverse)
library(mdsr)
```

Section 5.2.4: #1, #3 and #4 only

QUESTION 1:

```
#1
(a <- filter(flights, arr_delay >= 120))

## # A tibble: 10,200 x 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 <dtm>

#2
(b <- filter(flights, dest %in% c('IAH', 'HOU')))

## # A tibble: 9,313 x 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 <dtm>
```

```
#3
```

```
(c <- filter(flights, carrier %in% c('United','American','Delta')))
```

```
## # A tibble: 0 x 19
```

```
## # ... with 19 variables: year <int>, month <int>, day <int>,  
## #   dep_time <int>, sched_dep_time <int>, dep_delay <dbl>, arr_time <int>,  
## #   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 <dtm>
```

```
#4
```

```
(d <- filter(flights, month == 7 | month == 8 | month == 9))
```

```
## # A tibble: 86,326 x 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 <dtm>
```

```
#5
```

```
(e <- filter(flights, arr_delay >= 120, dep_delay <= 0))
```

```
## # A tibble: 29 x 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 <dtm>
```

```
#6
```

```
(f <- filter(flights, dep_delay >= 60, dep_delay - arr_delay >= 30))
```

```
## # A tibble: 2,074 x 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 <dtm>
```

```
#7
(g <- filter(flights, dep_time >= 0, dep_time <= 600))
```

```
## # A tibble: 9,344 x 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 <dtm>
```

QUESTION 3:

```
# find dep_time having missing values
(dep_time_missing <- filter(flights, is.na(dep_time)))
```

```
## # A tibble: 8,255 x 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 8,245 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 <dtm>
```

ANSWER 3:

From the above dataset, the rows with missing dep_time are also missing the values for dep_delay and arr_time variables, that means they never departed and arrived which are only planned flights.

QUESTION 4:

```
(x <- NA ^ 0)

## [1] 1

(y <- NA * TRUE)

## [1] NA

(z <- FALSE - NA)

## [1] NA
```

ANSWER 4:

In general, any mathematical operation with a missing value results another missing value and we need to ask explicitly for missing values in case of conditions.

Section 5.4.1: #1 and #3 only

QUESTION 1: