Pre_test_Team_GA

2022-10-06

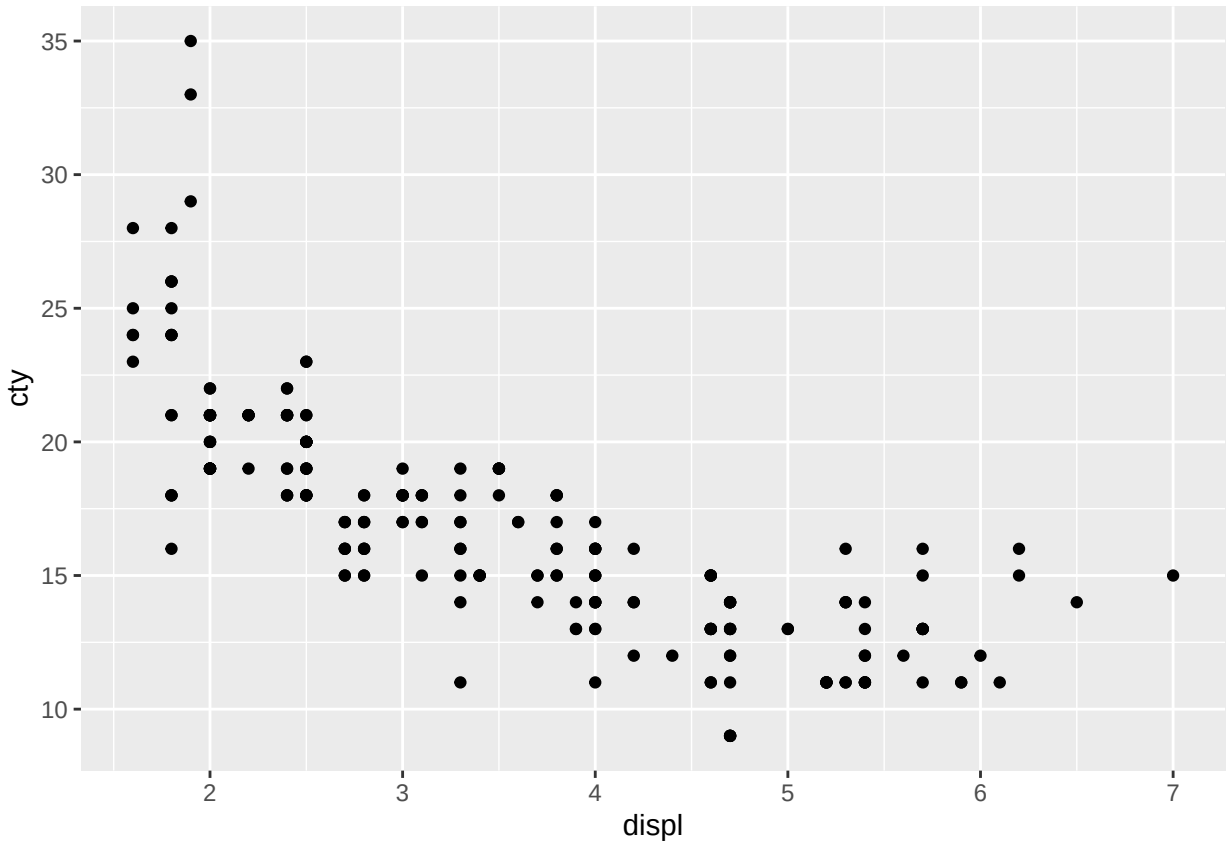
```
  412/612  PRACTICE TEST
PART 1  GGPLOT PRACTICE  (Use tidyverse coding for all problems)
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

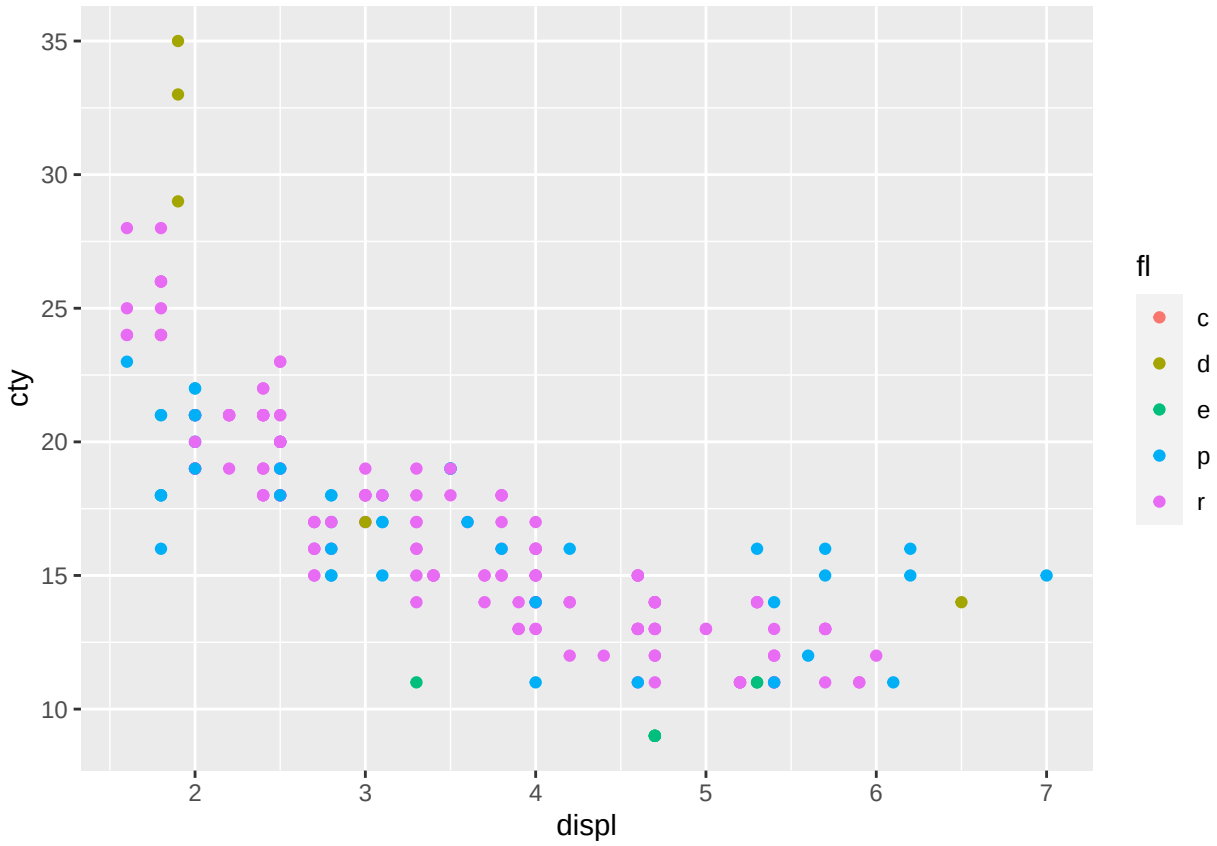
1) Using the mpg data frame, create a scatter plot that shows a relationship between the variables displ and cty. (displ =x and cty=y)

```
#data(mpg)
ggplot(data=mpg, aes(x=displ, y=cty))+
geom_point()
```



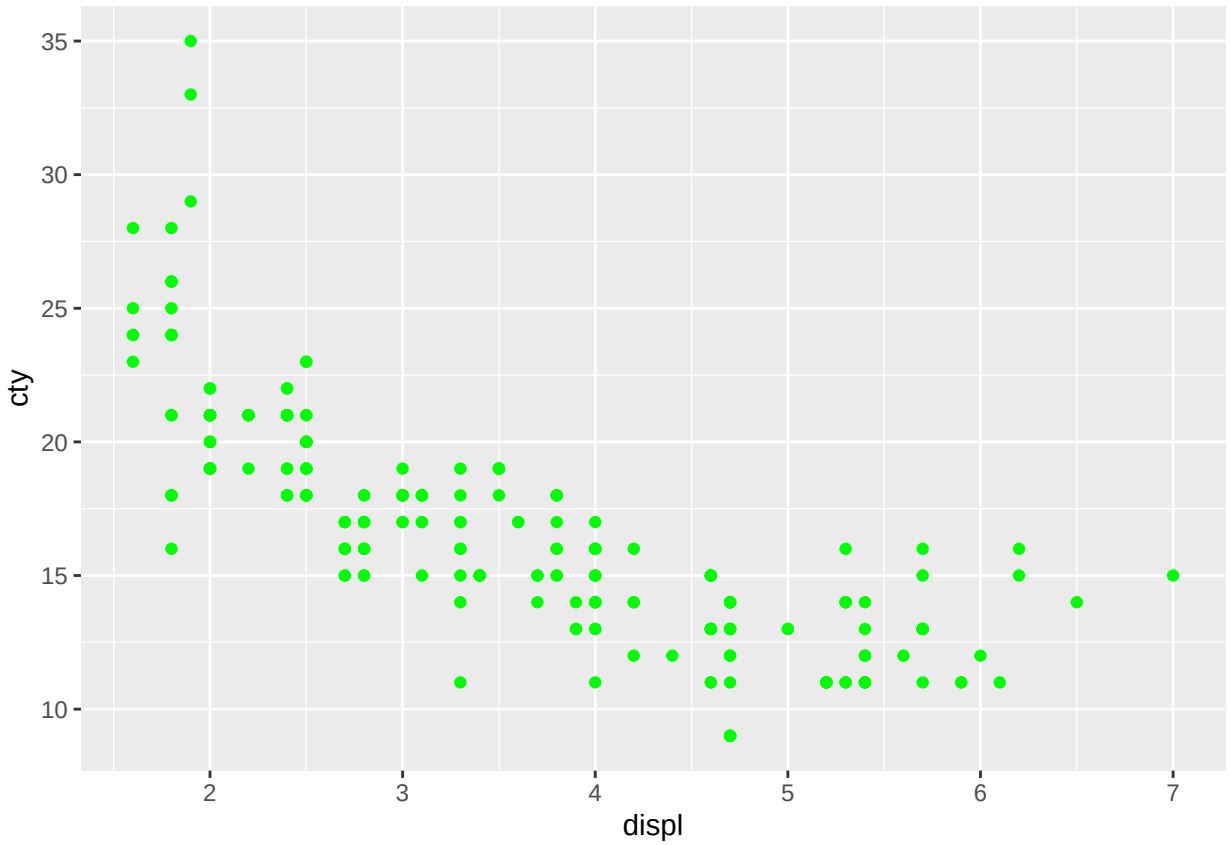
2 Using the mpg data frame, create a scatter plot that shows a relationship between the variables displ and cty. (displ =x and cty=y),and also map colors of your scatter plot to the variable fl.

```
ggplot(data=mpg, aes(x=displ, y=cty, color =fl))+
geom_point()
```



3. Using the mpg data frame, create a scatter plot that shows a relationship between the variables displ and cty. (displ =x and cty=y), and also include code so that all of your scatter plot points are green.

```
ggplot(data=mpg, aes(x=displ, y=cty))+
geom_point(color= 'green')
```

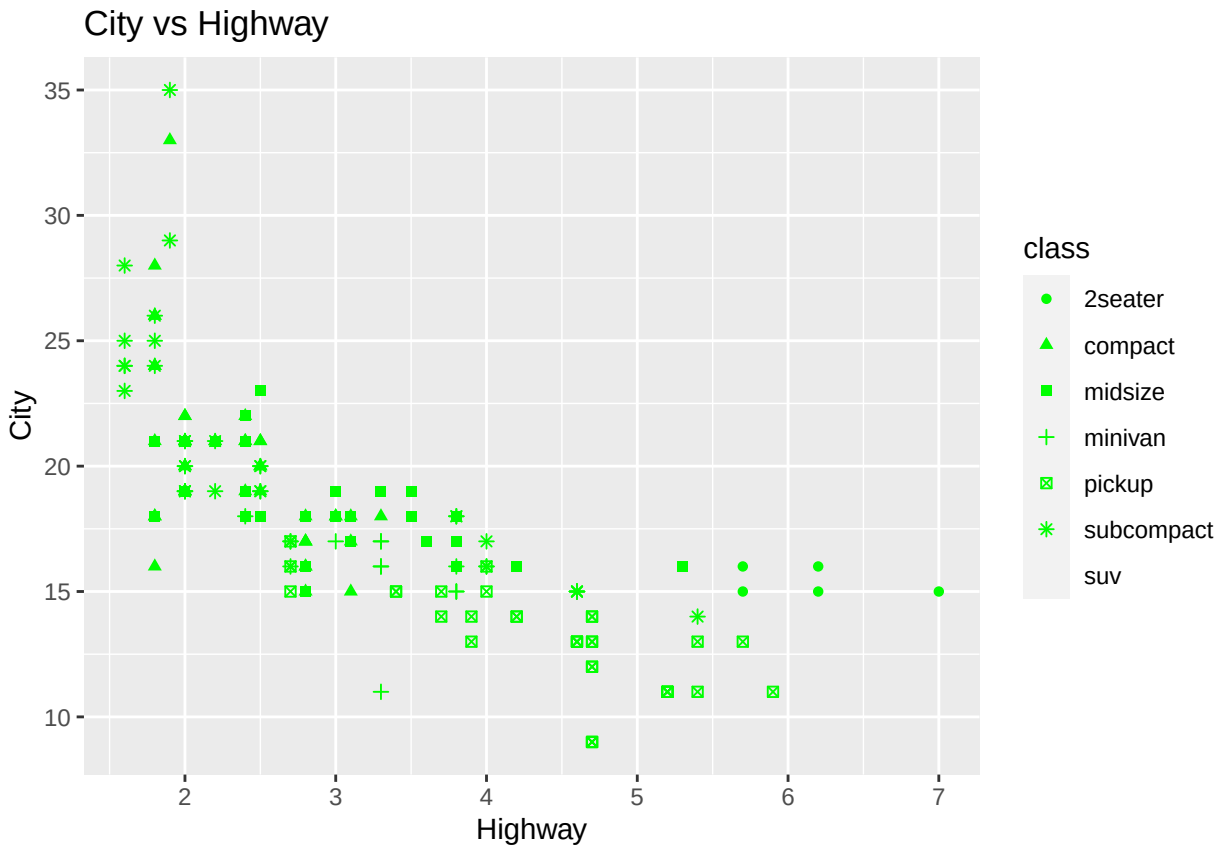


4. Using the mpg data frame, create a scatter plot that shows a relationship between the variables displ and cty. (displ =x and cty=y), and also include code so that the scatter plot has different shapes or characters according to class. Code so that your scatter plot has the title City vs Highway , the y axis is labled City and the x axis is labeld Highway. (Check out the ggplot graphing example towards the end of chapter one in your book to get data points with different shapes)

```
ggplot(data=mpg, aes(x=displ, y=cty, shape= class))+
xlab("Highway") +
ylab("City") +
ggtitle("City vs Highway")+
geom_point(color='green')
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 7. Consider
## specifying shapes manually if you must have them.
```

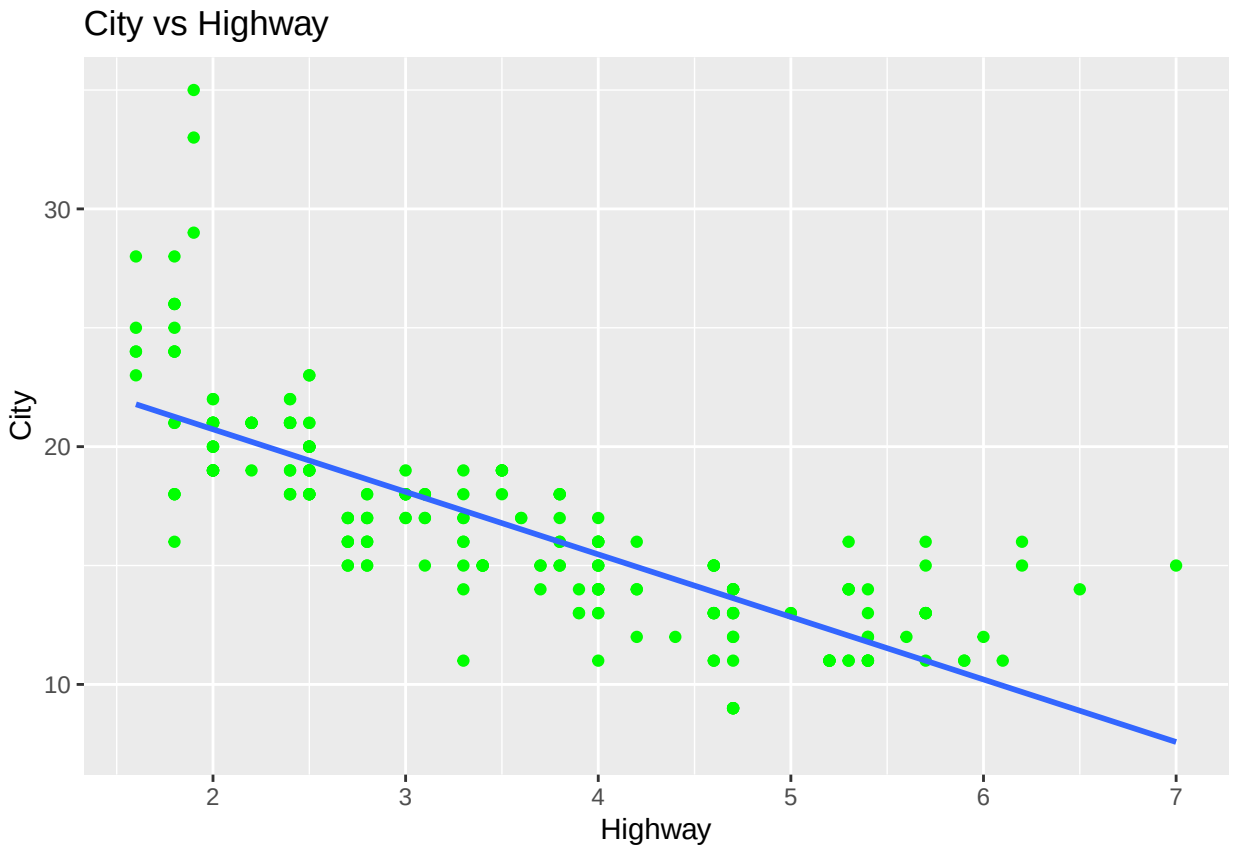
```
## Warning: Removed 62 rows containing missing values (geom_point).
```



5. Using the mpg data frame, create a smooth line fitted to the data displ and cty. (displ =x and cty=y).

```
ggplot(data=mpg, aes(x=displ, y=cty))+
xlab("Highway") +
ylab("City") +
ggtitle("City vs Highway")+
geom_point(color='green') +
geom_smooth(se = FALSE, method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



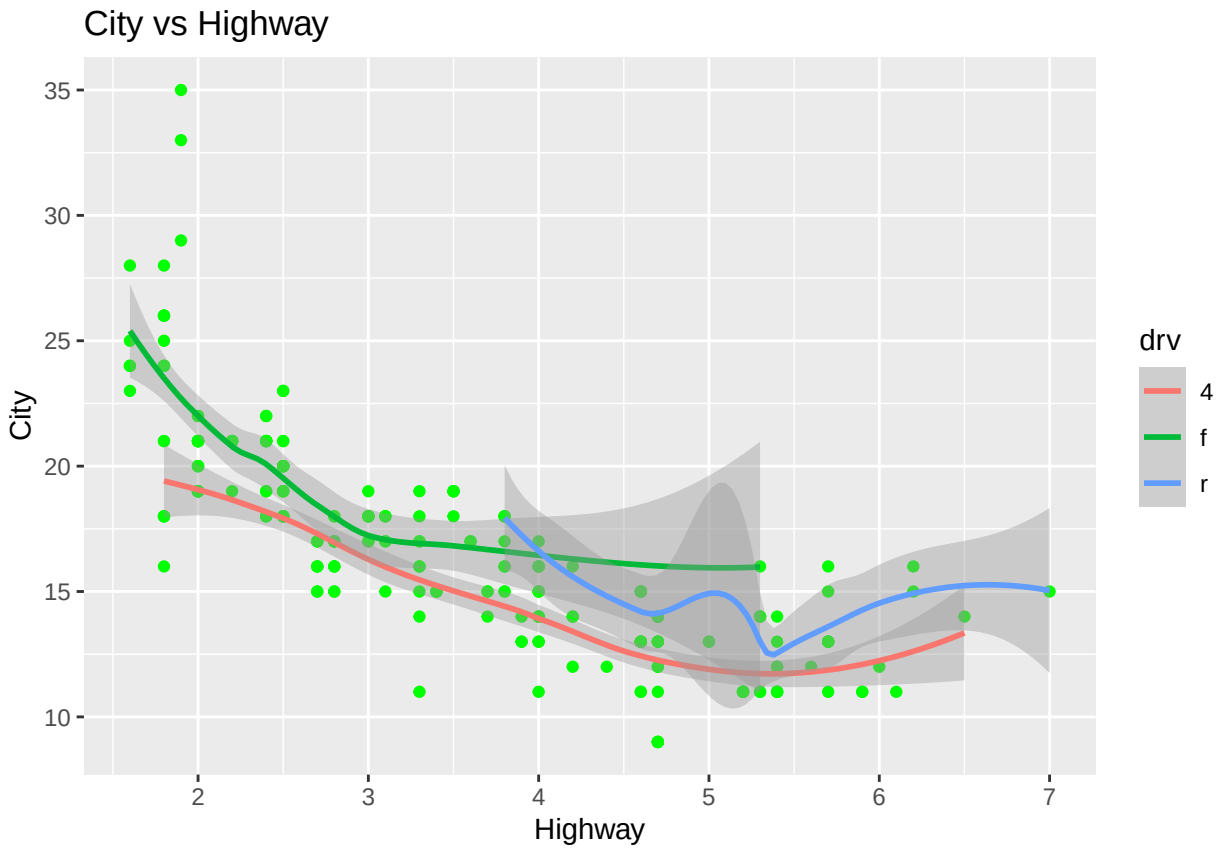

```
geom_smooth()
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## position_identity
```

6. Using the mpg data frame, create smooth lines over the scatter plot for the data displ and cty. (displ =x and cty=y) based on drv levels.

```
ggplot(data=mpg, aes(x=displ, y=cty, color=drv))+
xlab("Highway") +
ylab("City") +
ggtitle("City vs Highway")+
geom_point(color= "green") +
#geom_smooth(se = FALSE, method = lm)
geom_smooth()
```

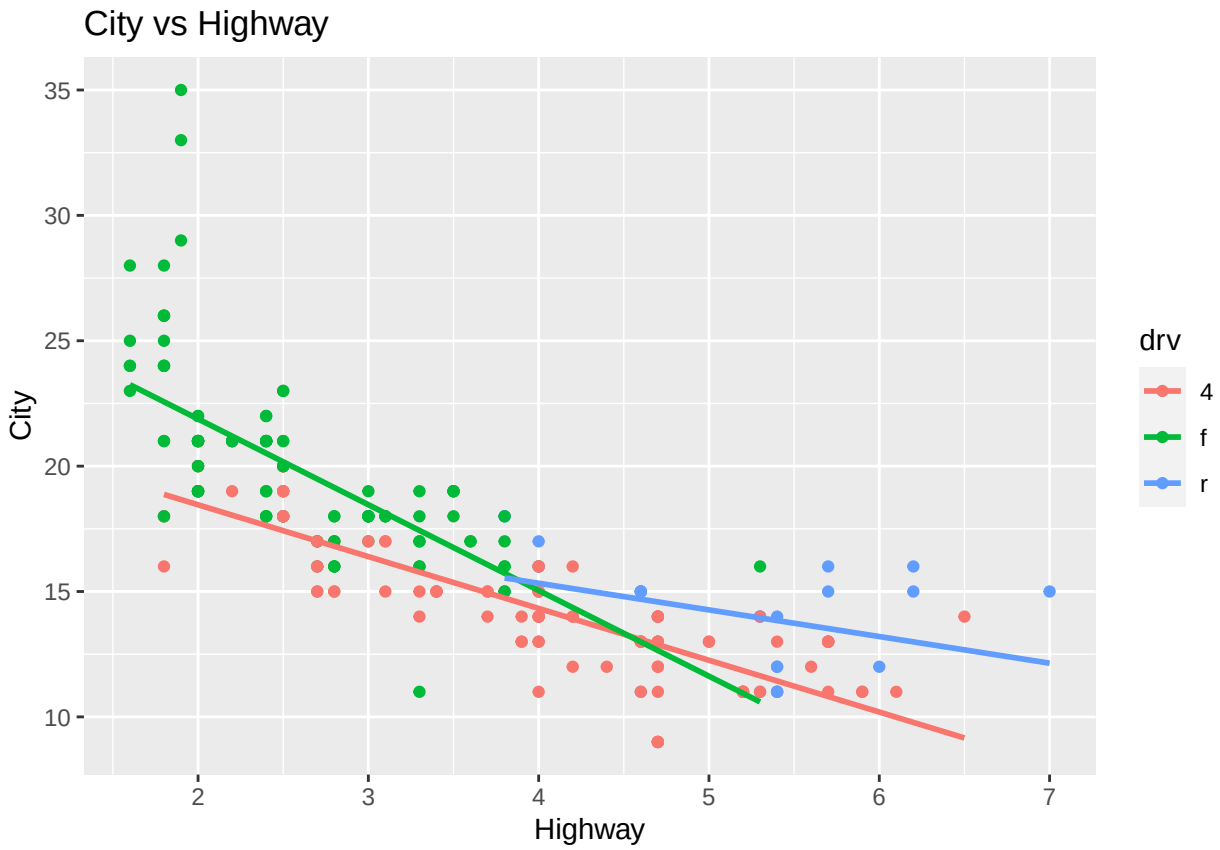
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



7. Using the mpg data frame, create a scatter plot that shows a relationship between the variables displ and cty. (displ =x and cty=y), and also include code that produces overlayed regression lines based on drv levels. The data points should also be color coded according to levels of drv.

```
ggplot(data=mpg, aes(x=displ, y=cty, color=drv))+
xlab("Highway") +
ylab("City") +
ggtitle("City vs Highway")+
geom_point() +
geom_smooth(se = FALSE, method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

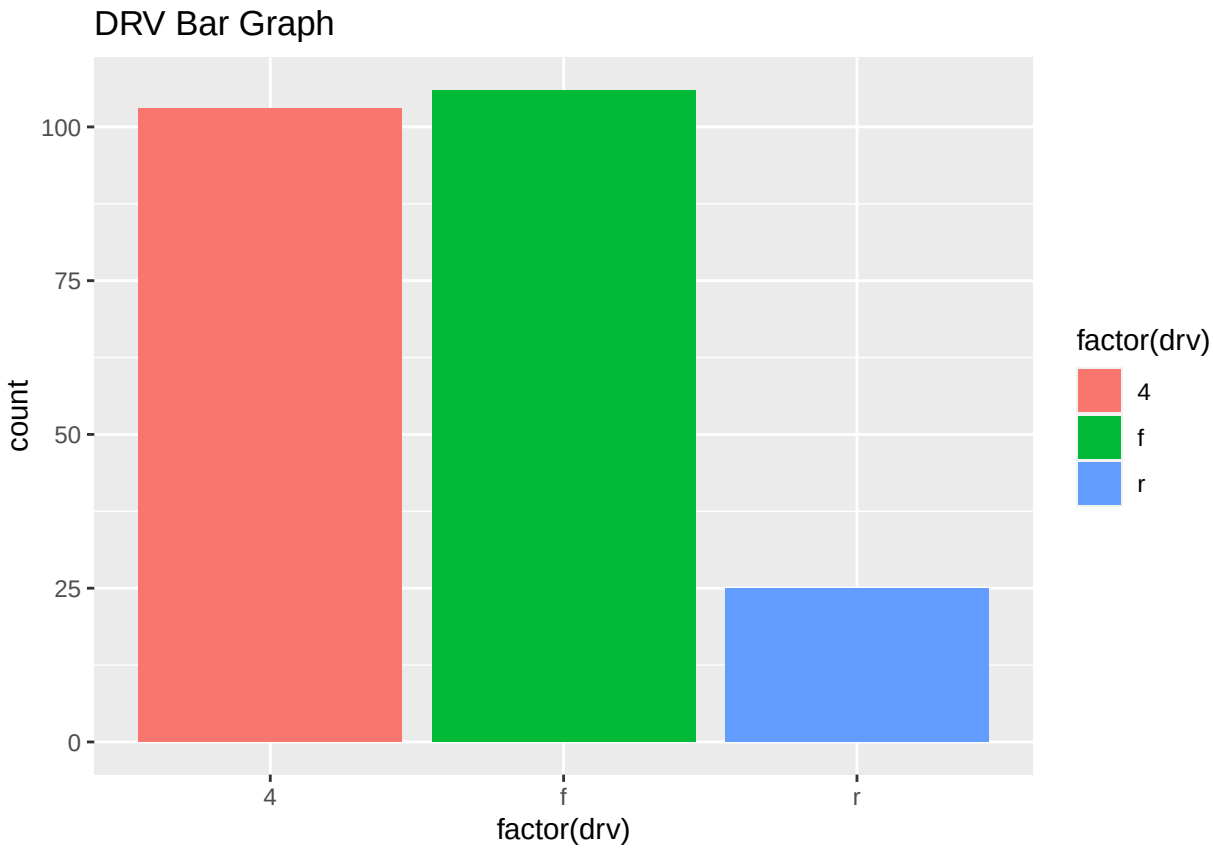



```
geom_smooth()
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## position_identity
```

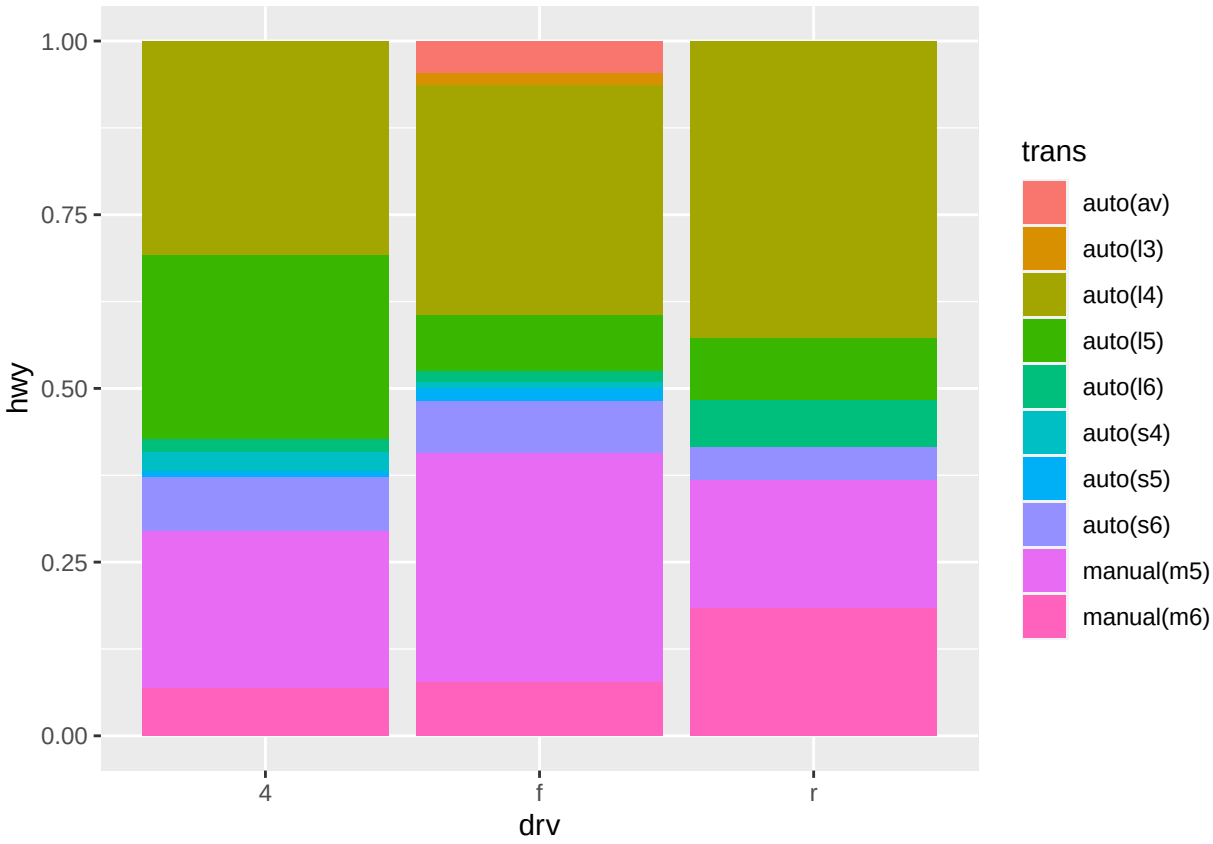
8. Using the mpg data frame, create a bar graph for the variable drv. include code so that each bar has a different color. Title your bar graph DRV Bar Graph

```
ggplot(data=mpg, aes(factor(drv), fill = factor(drv))) +
ggtitle("DRV Bar Graph")+
geom_bar()
```



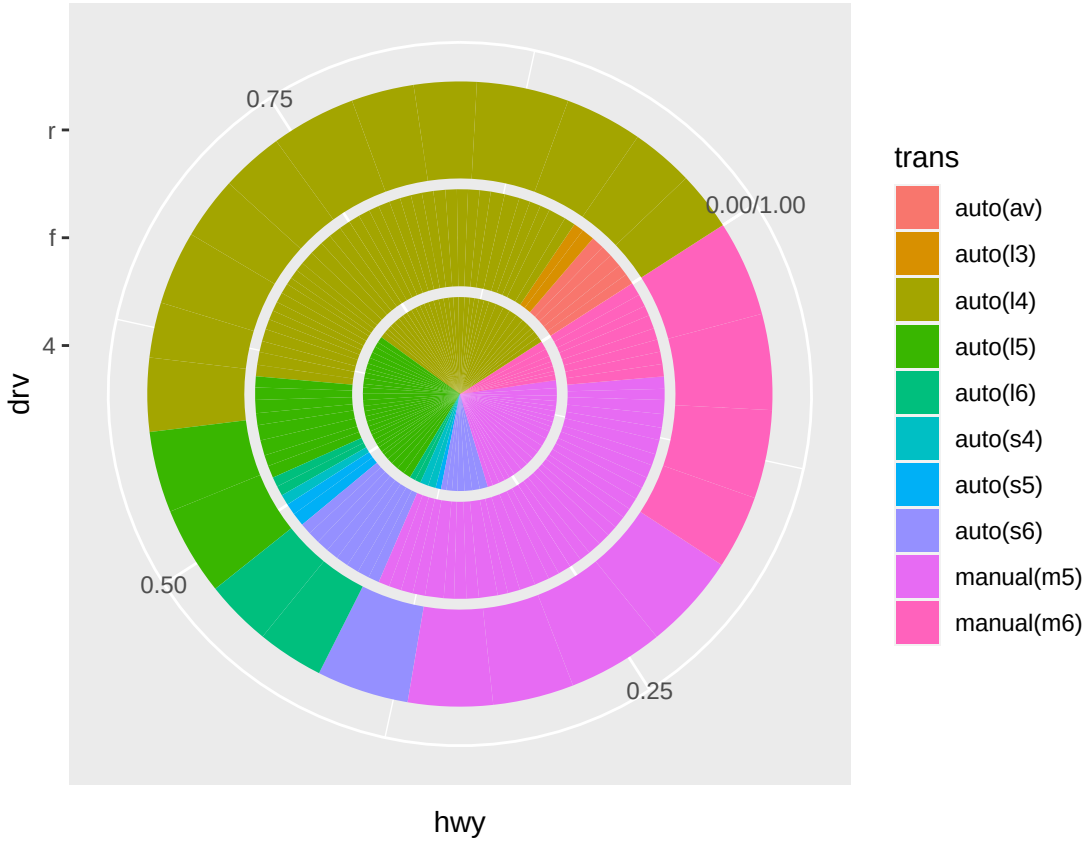
9. Using the mpg data frame, for the variable drv, create a bar graph that shows color stacked bars over the variable trans for the variable hwy

```
ggplot(mpg, aes(x = drv, y = hwy, fill = trans)) +  # Create stacked bar chart
  geom_bar(stat = "identity", position = 'fill')
```



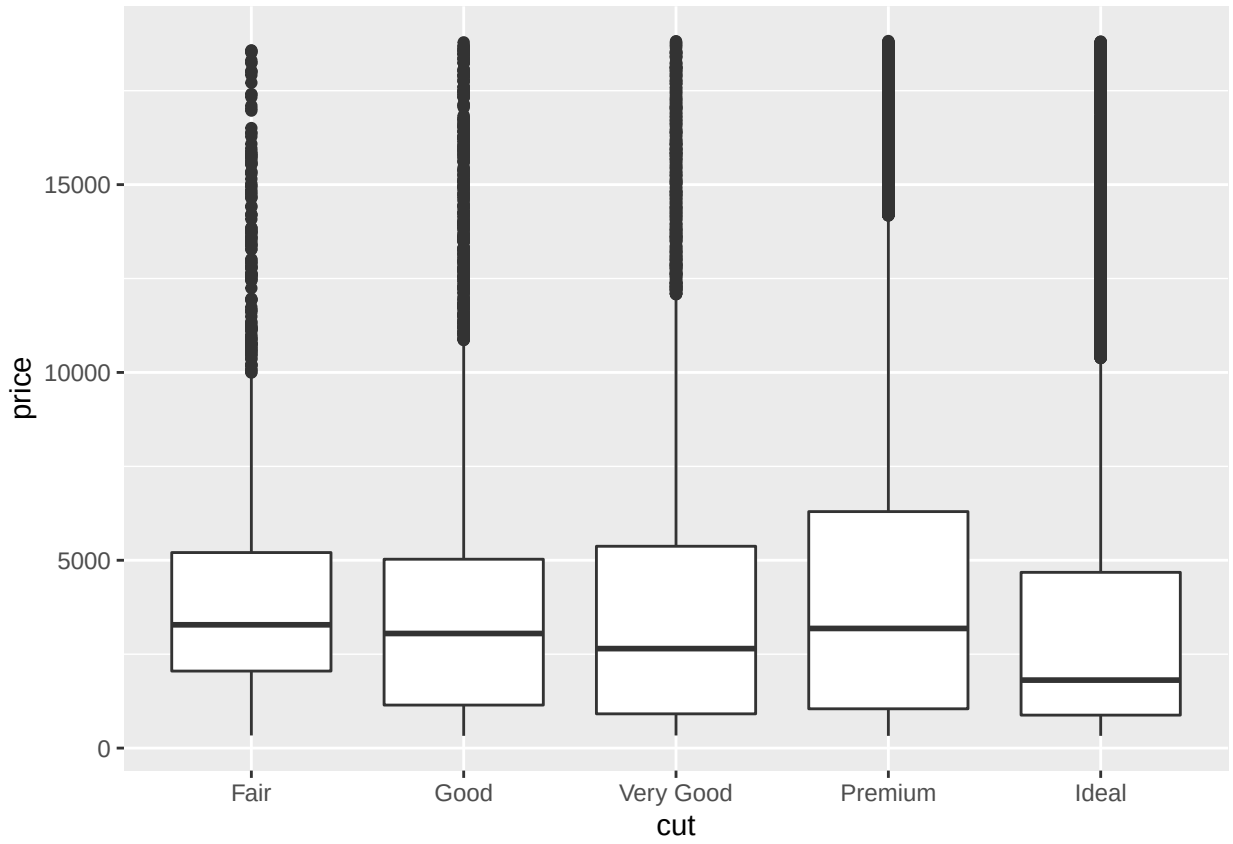
10. Using the stacked bar graph from number 10, create a pie chart.

```
ggplot(mpg, aes(x = drv, y = hwy, fill = trans)) +  # Create stacked bar chart
geom_bar(stat = "identity", position= "fill") +
coord_polar("y", start=1)
```



11. Use and show R code that will produce the following side by side box plots from the diamonds data frame.

```
data("diamonds")
ggplot(diamonds, aes(y= price, x= cut))+
  geom_boxplot()
```



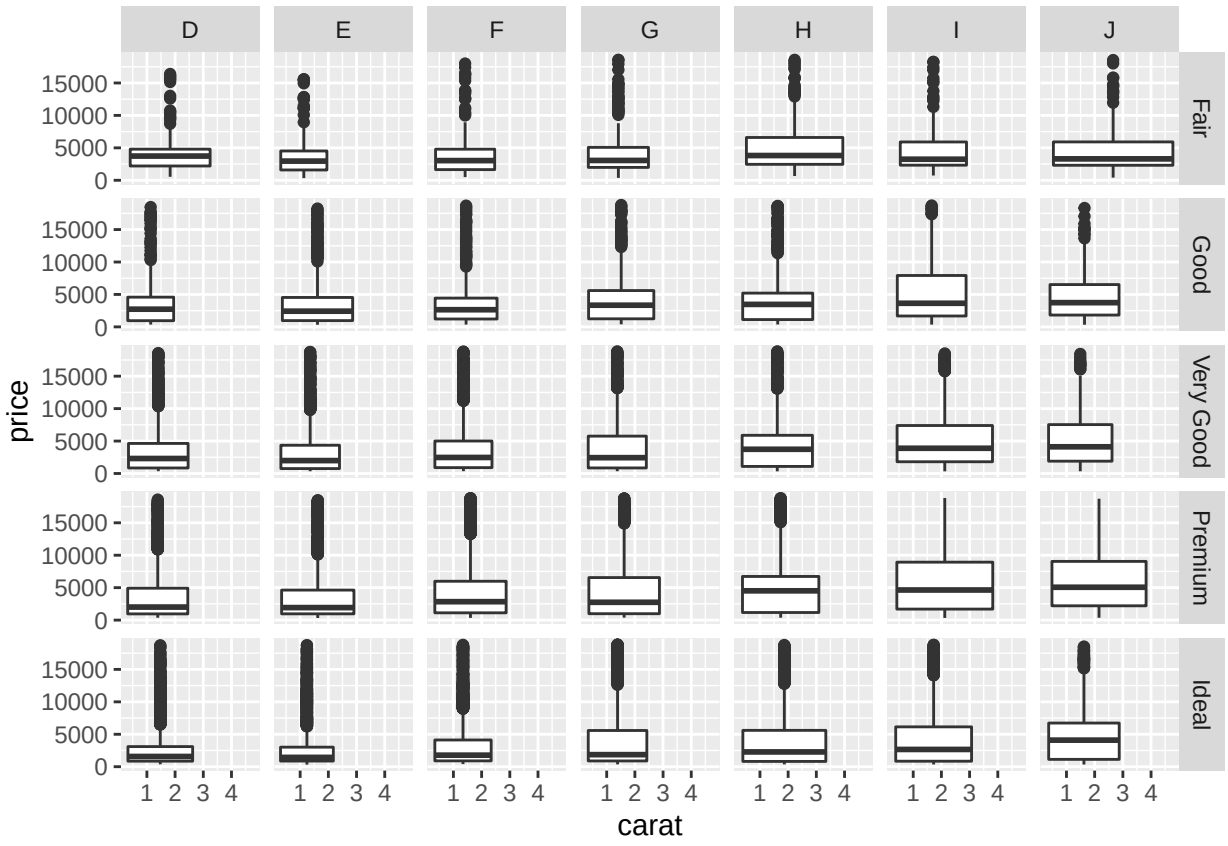

```
diamonds
```

```
## # A tibble: 53,940 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
## # ... with 53,930 more rows
```

12. Use and show R code that will produce the following faceted display of boxplots from the diamonds data frame.

```
ggplot(data = diamonds, aes(x=carat, y= price)) +
geom_boxplot() +
facet_grid(cut~color)
```

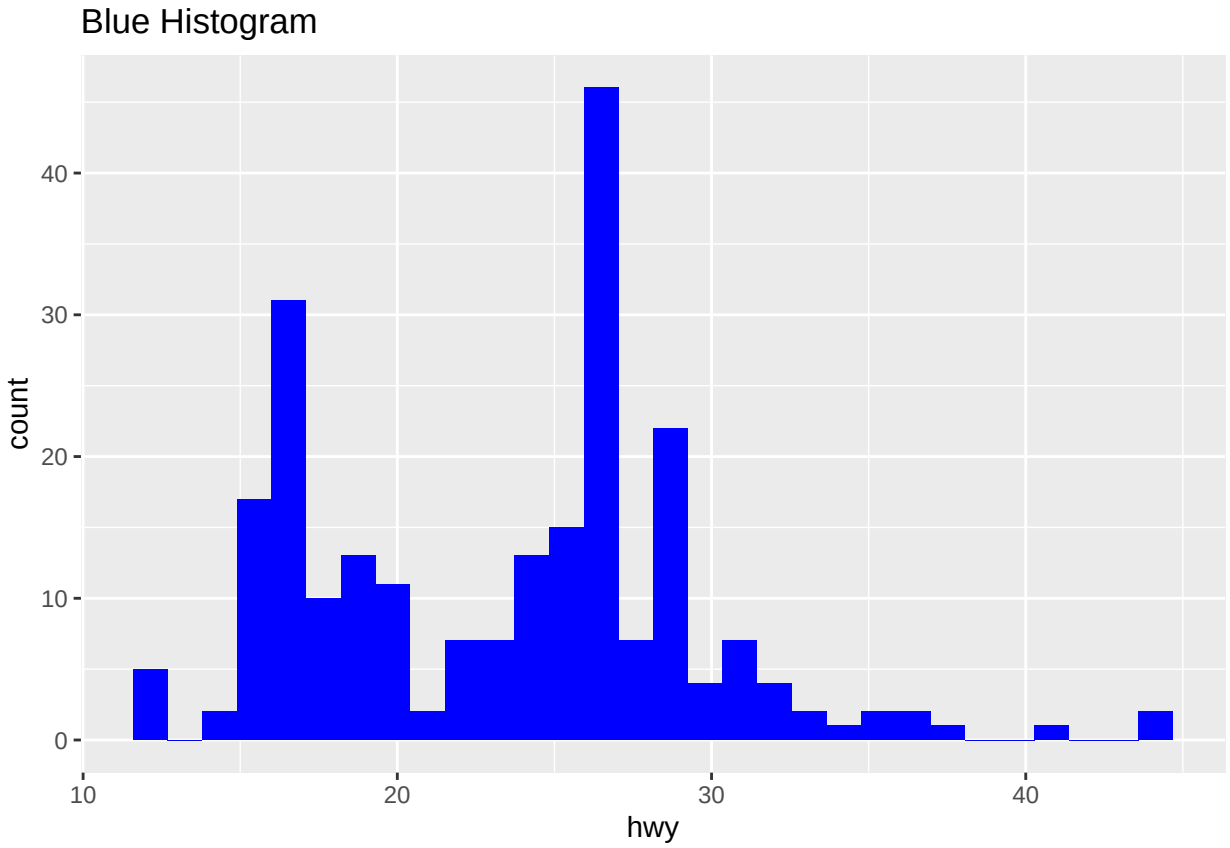
```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```



13. Use and show R code that will produce a histogram that is colored blue for the hwy variable from the mpg data frame. Give the histogram the title Blue Histogram.

```
ggplot(mpg, aes(x=hwy))+
  geom_histogram(fill = "blue")+
  ggtitle("Blue Histogram")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



PART II DPLYR PRACTICE (You may use the piping method or the assignment method)

1. Using the DPLYR filter function and the mpg data frame, produce a data frame that only has output for a Dodge Durango 4wd.

```
library(dplyr)

Dodge_durango_4wd<- mpg %>% filter(manufacturer == "dodge" & model == "durango 4wd")

Dodge_durango_4wd
```

```
## # A tibble: 7 x 11
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 dodge        durango 4wd   3.9  1999     6 auto~ 4        13    17 r     suv
## 2 dodge        durango 4wd   4.7  2008     8 auto~ 4        13    17 r     suv
## 3 dodge        durango 4wd   4.7  2008     8 auto~ 4         9    12 e     suv
## 4 dodge        durango 4wd   4.7  2008     8 auto~ 4        13    17 r     suv
## 5 dodge        durango 4wd   5.2  1999     8 auto~ 4        11    16 r     suv
## 6 dodge        durango 4wd   5.7  2008     8 auto~ 4        13    18 r     suv
## 7 dodge        durango 4wd   5.9  1999     8 auto~ 4        11    15 r     suv
```

2. Using the DPLYR filter function and the mpg data frame, produce a data frame that only has output for vehicles whose city mileage is less than 10 miles per gallon and whose highway mileage is less than 16 miles per gallon.

```
vhicle_milage_cityunder10_highwayunder16<- mpg %>% filter(cty < 10  & hwy<16)
vhicle_milage_cityunder10_highwayunder16
```

```
## # A tibble: 5 x 11
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 dodge        dakota pic~   4.7  2008     8 auto~ 4         9    12 e     pick~
## 2 dodge        durango 4wd   4.7  2008     8 auto~ 4         9    12 e     suv
## 3 dodge        ram 1500 p~   4.7  2008     8 auto~ 4         9    12 e     pick~
## 4 dodge        ram 1500 p~   4.7  2008     8 manu~ 4         9    12 e     pick~
## 5 jeep         grand cher~   4.7  2008     8 auto~ 4         9    12 e     suv
```

3. Using the DPLYR arrange function and the mpg data frame, produce a data frame that displays displ in descending order

```
displ_desc<-mpg %>% arrange(displ)

head(data.frame(displ_desc$displ))
```

```
##   displ_desc.displ
## 1              1.6
## 2              1.6
## 3              1.6
## 4              1.6
## 5              1.6
## 6              1.8
```

4. Using the DPLYR arrange function and the mpg data frame, produce a data frame of 30 observations that display city miles per gallon in ascending order. Which vehicle has the lowest city miles per gallon ?

```
p<-mpg %>% select(model, cty)%>%
  arrange(cty, desc= FALSE) # arrange city miles per gallon in ascending order

head(p,30) # shows the first 30 rows
```

```
## # A tibble: 30 x 2
##    model                 cty
##    <chr>               <int>
##  1 dakota pickup 4wd       9
##  2 durango 4wd             9
##  3 ram 1500 pickup 4wd     9
##  4 ram 1500 pickup 4wd     9
```

```
##  5 grand cherokee 4wd      9
##  6 c1500 suburban 2wd     11
##  7 k1500 tahoe 4wd        11
##  8 k1500 tahoe 4wd        11
##  9 caravan 2wd            11
## 10 dakota pickup 4wd      11
## # ... with 20 more rows
```

```
head(p,1) #dodge with model dakota pickup 4wd vehicle has the lowest city miles per gallon
```

```
## # A tibble: 1 x 2
##   model               cty
##   <chr>             <int>
## 1 dakota pickup 4wd     9
```

#5. Using the DPLYR filter and select functions and the mpg data frame, produce a data frame that displays all ford vehicles for 1999 whose city miles per gallon is less than 16 and whose highway miles per gallon is also less than 16.

```
p_ford1999<-mpg %>% filter(year==1999 & manufacturer== "ford" & cty < 16 & hwy <16 )
p_ford1999 #only one vhicles found
```

```
## # A tibble: 1 x 11
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 ford         f150 picku~   5.4  1999     8 auto~ 4        11    15 r     pick~
```

#6. Using the DPLYR filter, select and mutate functions and the mpg data frame, produce a data frame that displays the difference between highway mileage and city mileage for the ford mustang.A partial data frame is given below; manufacturer model cty hwy difference 1 ford mustang 18 26 8 2 ford mustang 18 25 7 3 ford mustang 17 26 9

```
p_ford_mustang<- mutate(mpg, difference= hwy-cty) %>%
  filter(manufacturer=="ford" & model== "mustang") %>%
  select(manufacturer, model, cty, hwy, difference)

p_ford_mustang
```

```
## # A tibble: 9 x 5
##   manufacturer model     cty   hwy difference
##   <chr>        <chr>   <int> <int>      <int>
## 1 ford         mustang    18    26          8
## 2 ford         mustang    18    25          7
## 3 ford         mustang    17    26          9
## 4 ford         mustang    16    24          8
## 5 ford         mustang    15    21          6
## 6 ford         mustang    15    22          7
## 7 ford         mustang    15    23          8
## 8 ford         mustang    15    22          7
## 9 ford         mustang    14    20          6
```

#7. Install the New York City flights data package;install.packages("nycflights13") Now code and execute the following library library(nycflights.13) And then code and call the following data frame flights

```
library(nycflights13)
flights
```

```
## # A tibble: 336,776 x 19
```

```
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     1      517        515       2     830     819      11 UA
##  2  2013     1     1      533        529       4     850     830      20 UA
##  3  2013     1     1      542        540       2     923     850      33 AA
##  4  2013     1     1      544        545      -1    1004    1022     -18 B6
##  5  2013     1     1      554        600      -6     812     837     -25 DL
##  6  2013     1     1      554        558      -4     740     728      12 UA
##  7  2013     1     1      555        600      -5     913     854      19 B6
##  8  2013     1     1      557        600      -3     709     723     -14 EV
##  9  2013     1     1      557        600      -3     838     846      -8 B6
## 10  2013     1     1      558        600      -2     753     745       8 AA
## # ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

```
?flights
```

Use the following commands to further explore the data frame flights

```
names(flights)
```

```
##  [1] "year"           "month"          "day"            "dep_time"
##  [5] "sched_dep_time" "dep_delay"      "arr_time"       "sched_arr_time"
##  [9] "arr_delay"      "carrier"        "flight"         "tailnum"
## [13] "origin"         "dest"           "air_time"       "distance"
## [17] "hour"           "minute"         "time_hour"
```

Use DPLYR functions and the piping operator to produce a data frame that shows arrival delay times in descending order for American Airlines on March 17, 2013. A partial table is given below.

```
Americanairlines_172013_delay<-flights%>%
  filter(carrier == "AA" & month == 3 & day == 17 & year == 2013) %>%
  select(carrier, year, month, day, arr_delay)

arrange(Americanairlines_172013_delay, desc(arr_delay))
```

```
## # A tibble: 88 x 5
##    carrier  year month   day arr_delay
##    <chr>   <int> <int> <int>     <dbl>
##  1 AA       2013     3    17        67
##  2 AA       2013     3    17        39
##  3 AA       2013     3    17        39
##  4 AA       2013     3    17        36
##  5 AA       2013     3    17        33
##  6 AA       2013     3    17        22
##  7 AA       2013     3    17        22
```

```
##  8 AA       2013     3    17        21
##  9 AA       2013     3    17        19
## 10 AA       2013     3    17        19
## # ... with 78 more rows
```

#8. Using the mpg data frame , dplyr functions and the pipe operator, produce a data frame that displays the mean mpg for city driving for manufacturers in the year 1999 only in descending order. Which manufacturer got the best average gas mileage in 1999?

```
p<-mpg %>%
  filter(year==1999) %>%
  select(manufacturer, year, cty)%>%
  group_by(manufacturer) %>%
  summarize(avg=mean(cty))%>%
  arrange(desc(avg))
p  # Jeep got low avarge gas mile.
```

```
## # A tibble: 15 x 2
##    manufacturer   avg
##    <chr>        <dbl>
##  1 honda         24.8
##  2 volkswagen    21.2
##  3 subaru        19
##  4 hyundai       18.3
##  5 toyota        18.2
##  6 nissan        17.7
##  7 audi          17.1
##  8 pontiac       17
##  9 chevrolet     15.1
## 10 jeep          14.5
## 11 ford          13.9
## 12 mercury       13.5
## 13 dodge         13.4
## 14 land rover    11
## 15 lincoln       11
```