Class 05: Data Visualization with GGPLOT

Xiaoyan Wang(A16454066)

Q1. For which phases is data visualization important in our scientific workflows?

All of the above

Q2. True or False? The ggplot2 package comes already installed with R?

False

Q3. Which plot types are typically NOT used to compare distributions of numeric variables?

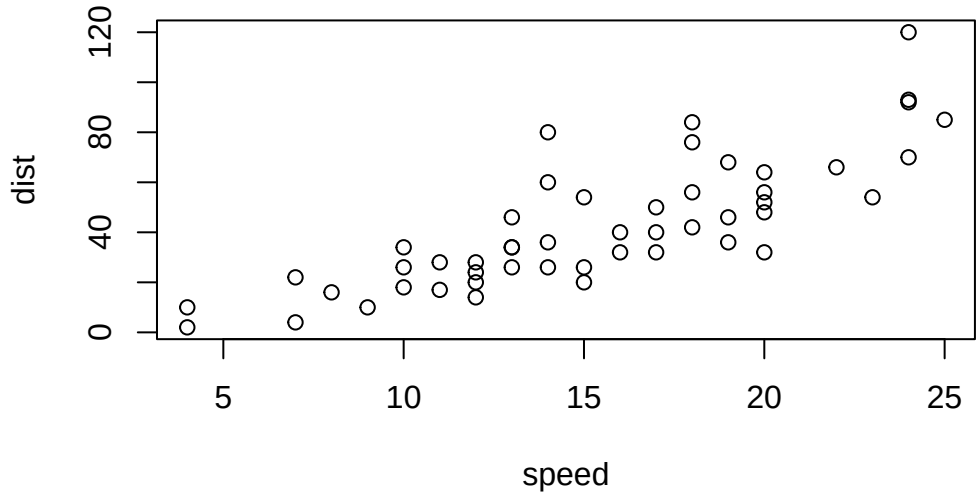
Network graphs

Q4. Which statement about data visualization with ggplot2 is incorrect?

ggplot2 is the only way to create plots in R

Plot in R

```
View(cars)
plot(cars)
```



Plot in ggplot

Specifing a dataset

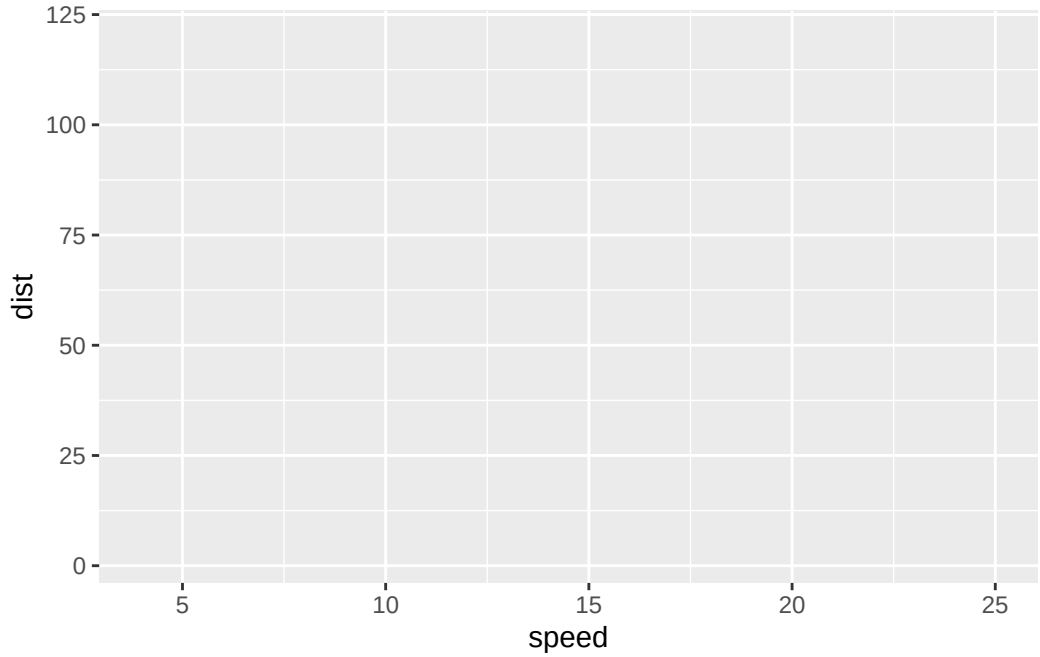
```
#install.packages("ggplot2")
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.3.3
```

```
ggplot(cars)
```

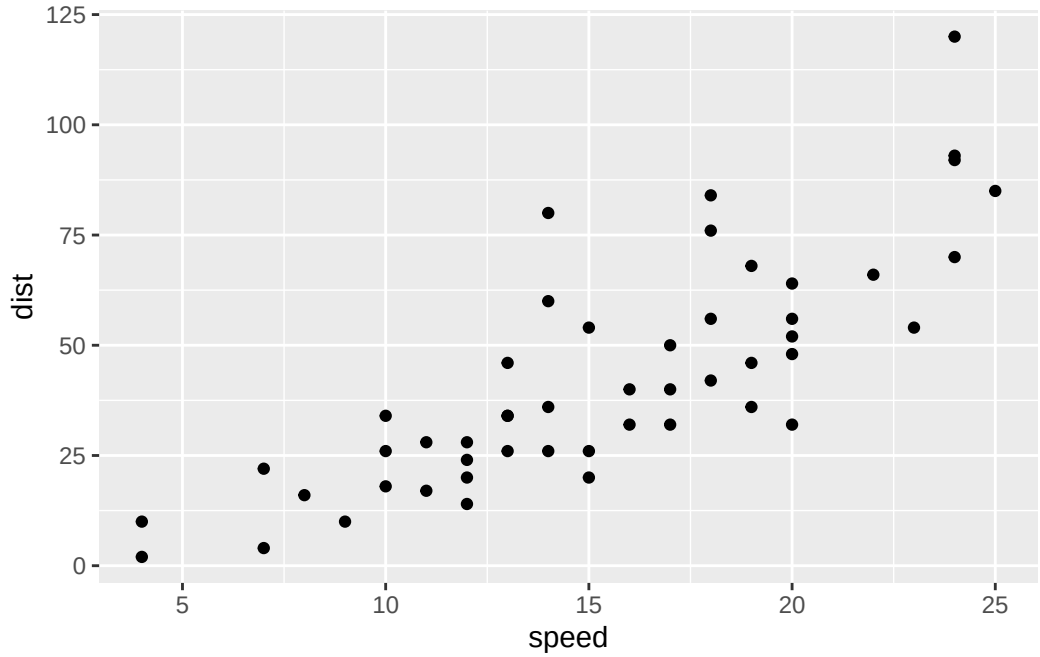
Specifing aesthetic mappings

```
ggplot(cars) +
  aes(x=speed, y=dist)
```



Specifing a geom layer

```
ggplot(data=cars) +
  aes(x=speed, y=dist) +
  geom_point()
```



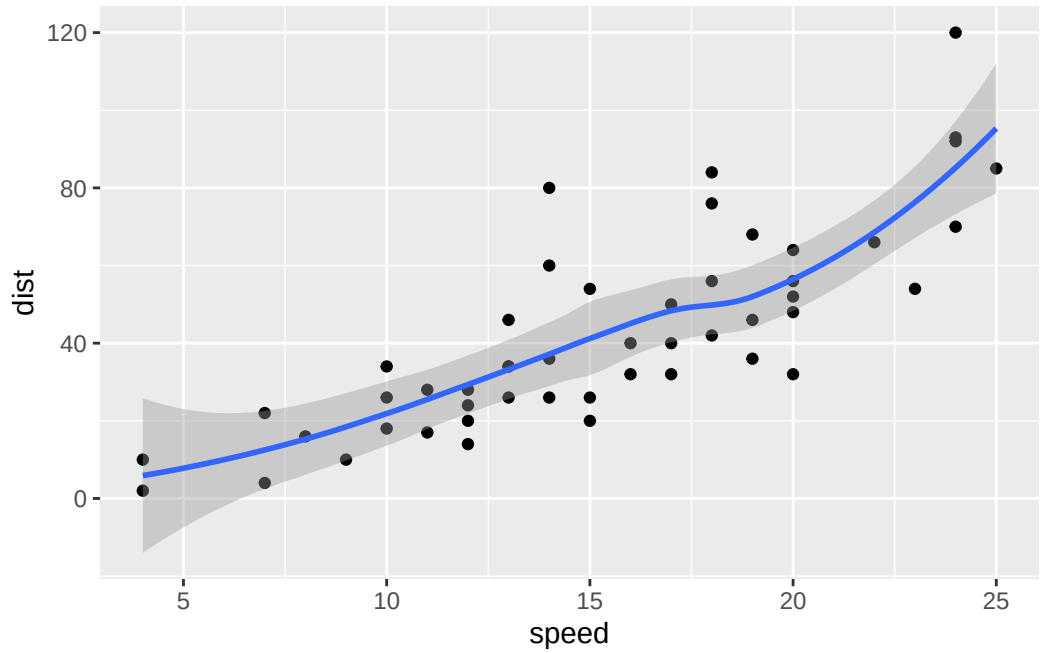
Q5. Which geometric layer should be used to create scatter plots in ggplot2?

geom_point()

Q6. In your own RStudio can you add a trend line layer to help show the relationship between the plot variables with the `geom_smooth()` function?

```
ggplot(data=cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  geom_smooth()
```

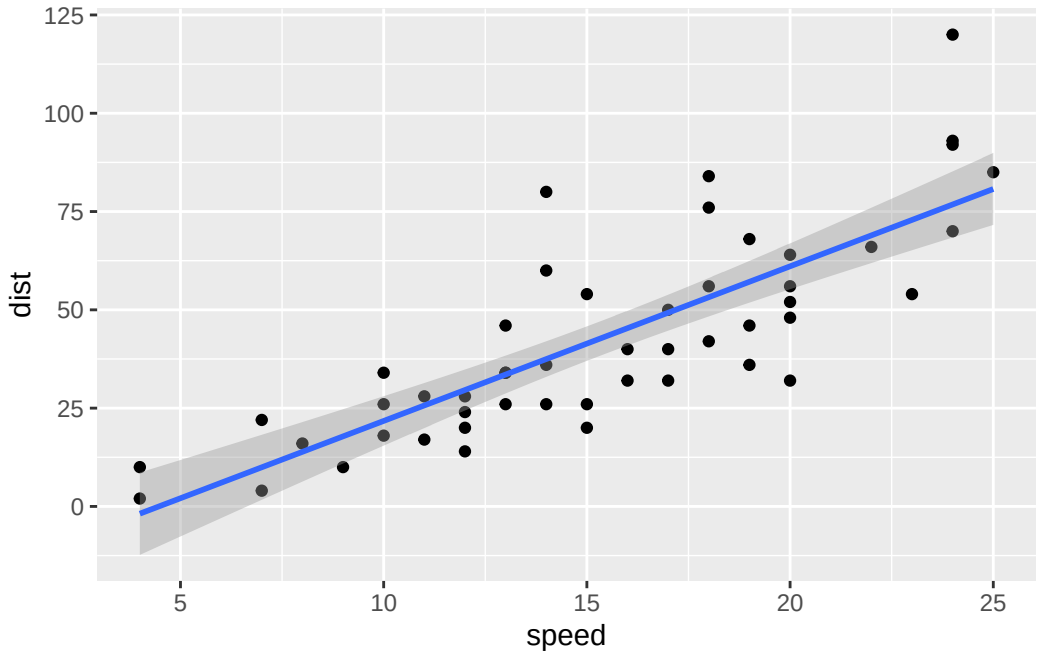
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Q7. Argue with `geom_smooth()` to add a straight line from a linear model without the shaded standard error region?

```
ggplot(data=cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  geom_smooth(method="lm")
```

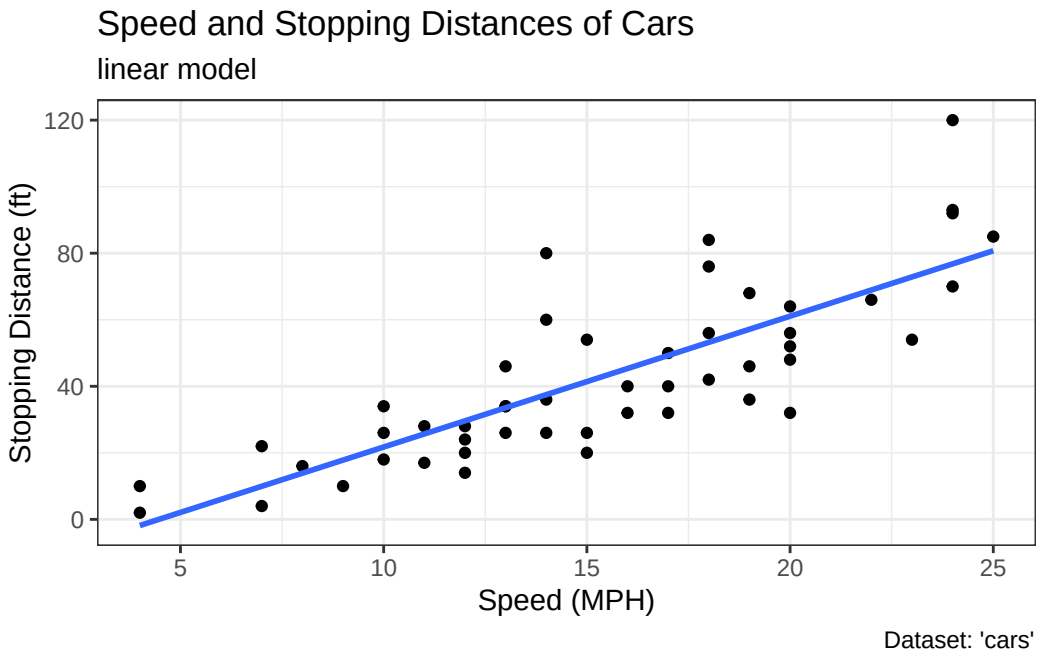
```
`geom_smooth()` using formula = 'y ~ x'
```



Q8. Can you finish this plot by adding various label annotations with the `labs()` function and changing the plot look to a more conservative "black & white" theme by adding the `theme_bw()` function:

```
ggplot(data=cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  labs(title="Speed and Stopping Distances of Cars",
       x="Speed (MPH)",
       y="Stopping Distance (ft)",
       subtitle = "linear model",
       caption="Dataset: 'cars'") +
  geom_smooth(method="lm", se=FALSE) +
  theme_bw()
```

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



Adding more plot aesthetics

```
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.txt"
genes <- read.delim(url)
head(genes)
```

```
        Gene Condition1 Condition2      State
1      A4GNT -3.6808610 -3.4401355 unchanging
2       AAAS  4.5479580  4.3864126 unchanging
3      AASDH  3.7190695  3.4787276 unchanging
4       AATF  5.0784720  5.0151916 unchanging
5       AATK  0.4711421  0.5598642 unchanging
6 AB015752.4 -3.6808610 -3.5921390 unchanging
```

Q9. Use the `nrow()` function to find out how many genes are in this dataset. What is your answer?

```
nrow(genes)
```

```
[1] 5196
```

Q10. Use the `colnames()` function and the `ncol()` function on the `genes` data frame to find out what the column names are (we will need these later) and how many columns there are. How many columns did you find?

4 columns were found

```
colnames(genes)
```

```
[1] "Gene"       "Condition1" "Condition2" "State"
```

```
ncol(genes)
```

```
[1] 4
```

Q11. Use the `table()` function on the `State` column of this data.frame to find out how many 'up' regulated genes there are. What is your answer?

127

```
table(genes$State)
```

```
      down unchanging         up
        72       4997        127
```

Q12. Using your values above and 2 significant figures. What fraction of total genes is up-regulated in this dataset?

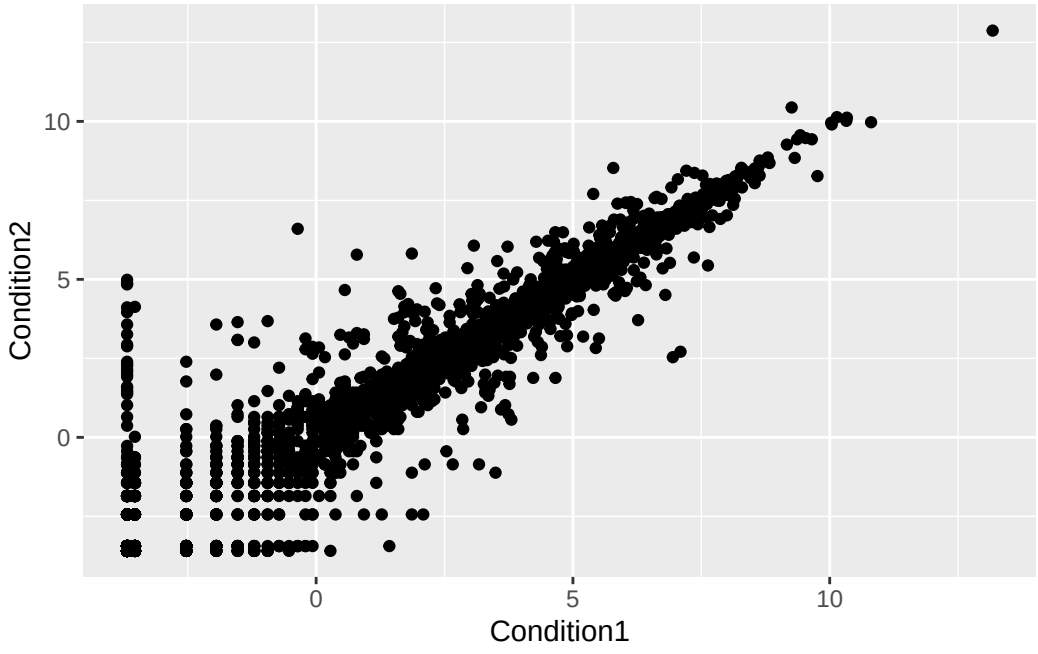
2.44%

```
round(table(genes$State)/nrow(genes)*100,2)
```

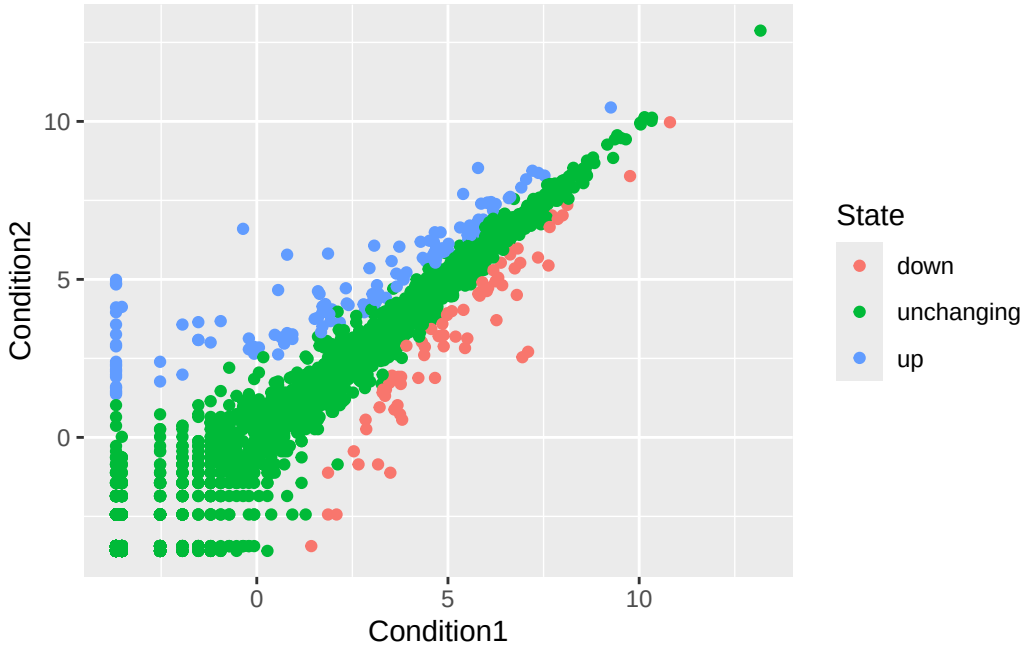
```
      down unchanging         up
      1.39      96.17       2.44
```

Q13. Complete the code below to produce the following plot

```
ggplot(genes) +
    aes(x=Condition1,y=Condition2) +
    geom_point()
```

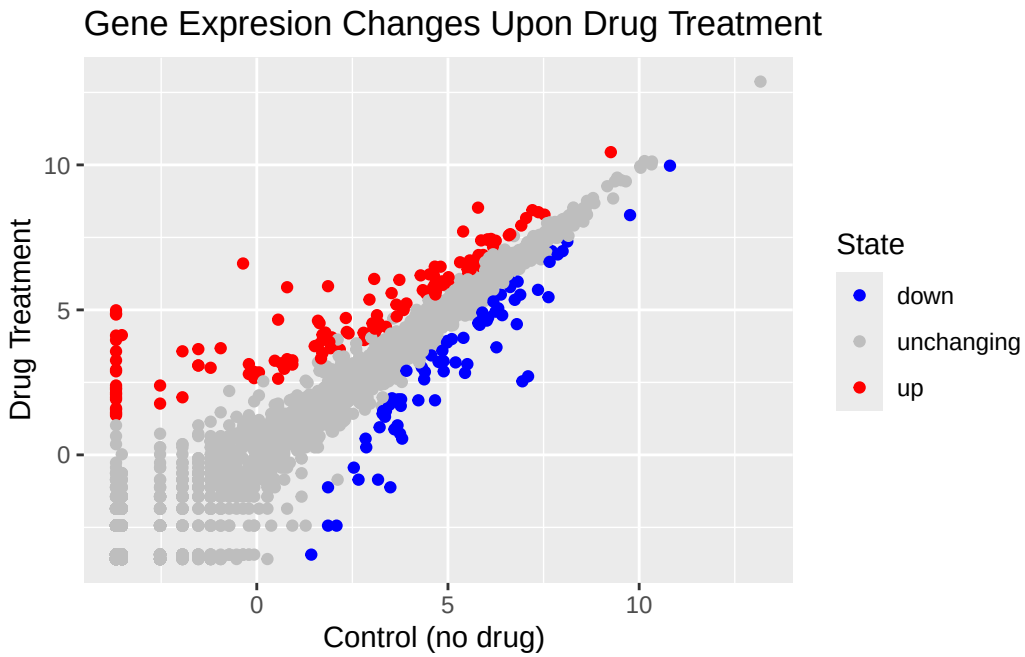



```
#Add some color
p <- ggplot(genes) +
    aes(x=Condition1, y=Condition2, col=State) +
    geom_point()
p
```



Q13. Nice, now add some plot annotations to the `p` object with the `labs()` function so your plot looks like the following:

```
p + scale_colour_manual(values=c("blue","gray","red"))+
  labs(title="Gene Expresion Changes Upon Drug Treatment",
         x="Control (no drug) ",
         y="Drug Treatment")
```



Going Further(optional)

```
# File location online
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.ts

gapminder <- read.delim(url)
```

```
# install.packages("dplyr")
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.3.3
```

```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

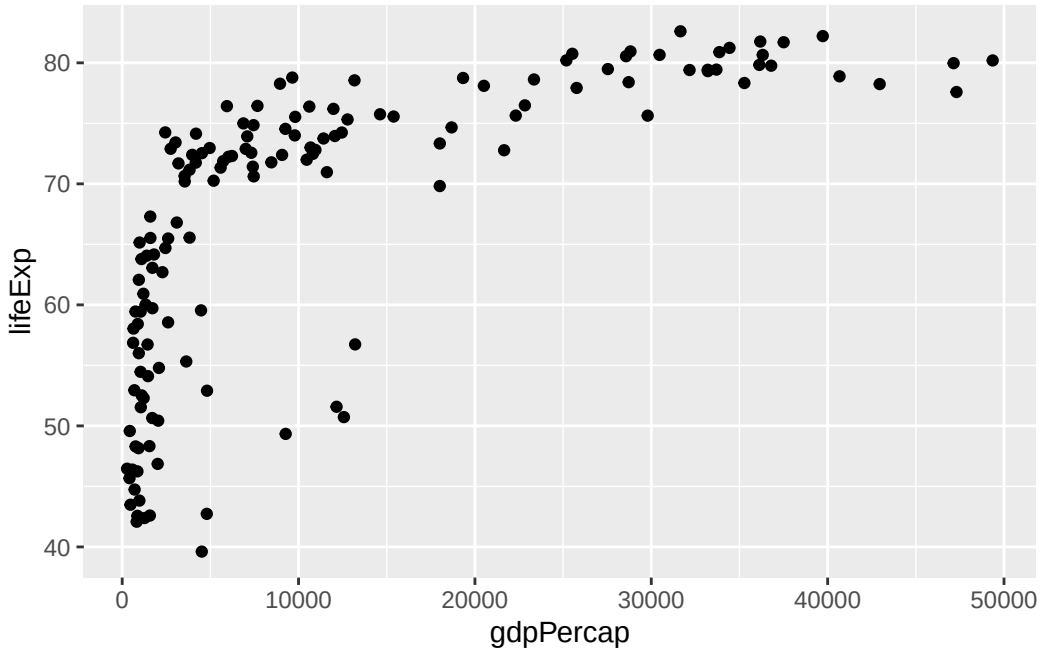
```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
gapminder_2007 <- gapminder %>% filter(year==2007)
```

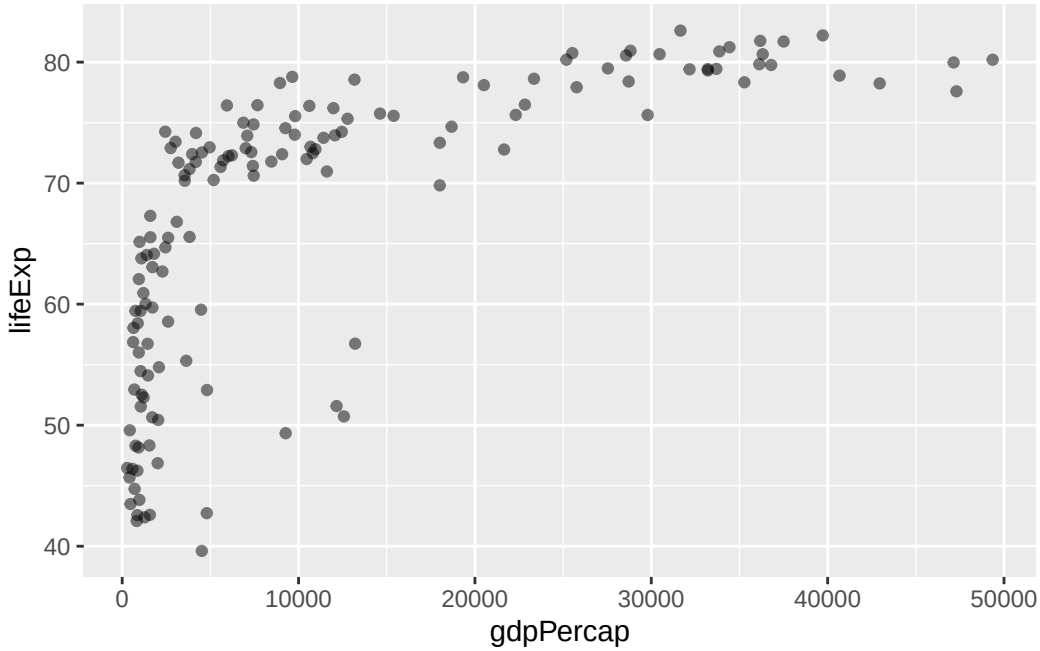
Let's consider the `gapminder_2007` dataset which contains the variables GDP per capita `gdpPercap` and life expectancy `lifeExp` for 142 countries in the year 2007

Q1. Complete the code below to produce a first basic scater plot of this `gapminder_2007` dataset:

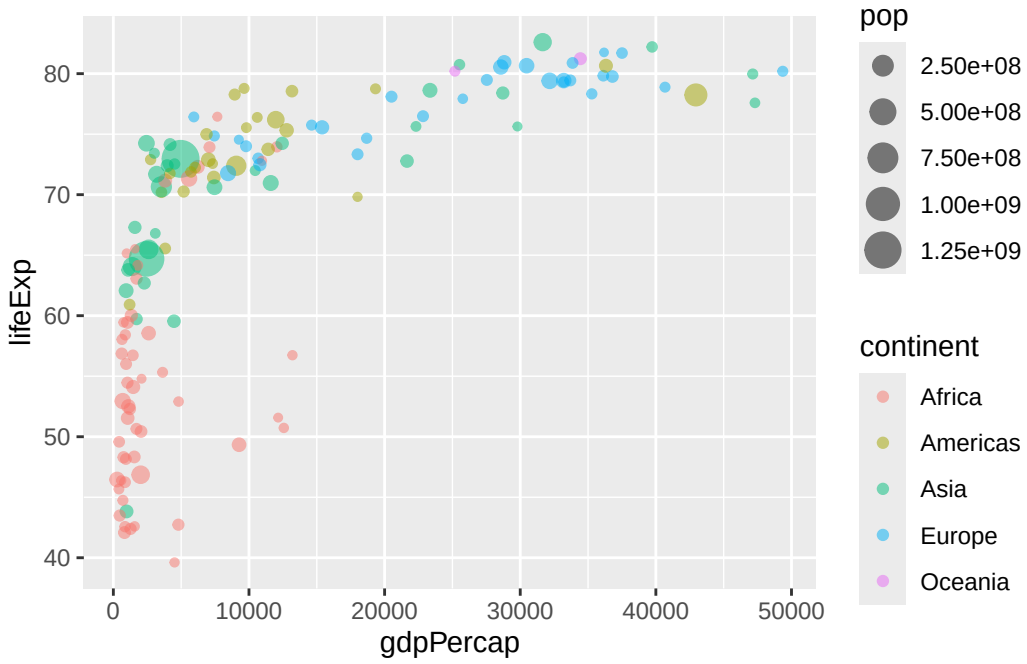
```
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +geom_point()
```



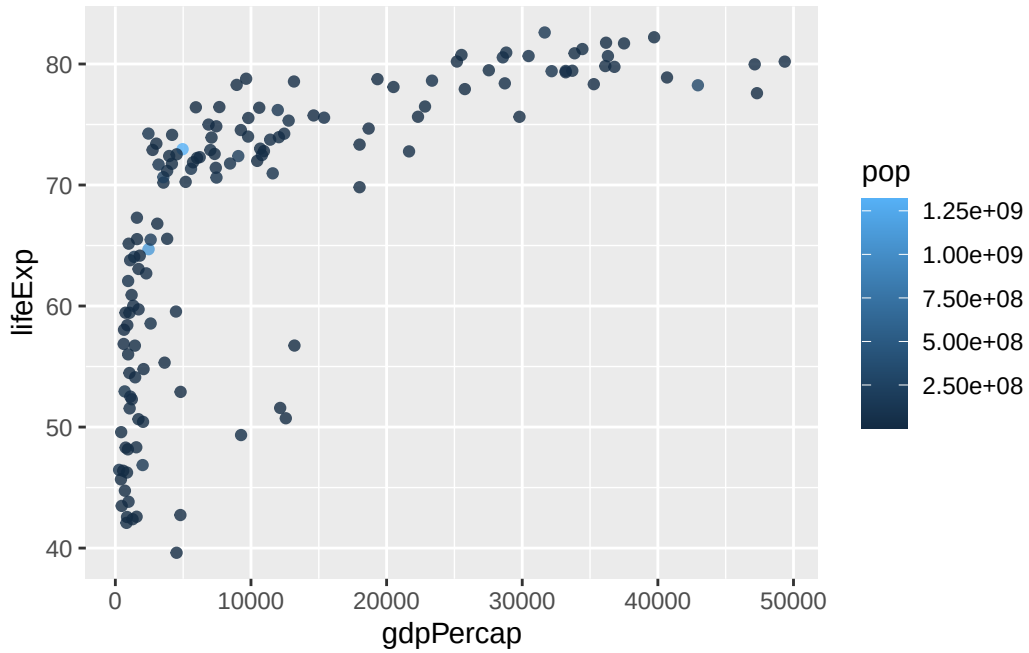

```
#Adjust points
g <- ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp)
g +geom_point(alpha=0.5)
```



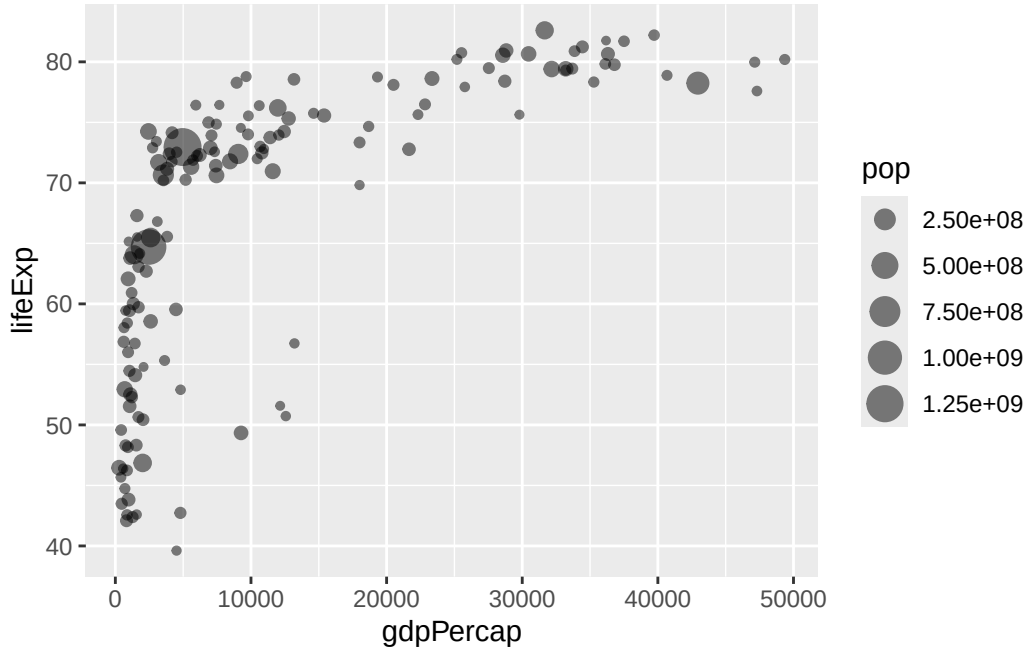

```
g+ aes(color=continent, size = pop)+geom_point(alpha=0.5)
```



```
g+ aes(color=pop)+geom_point(alpha=0.8)
```




```
g+ aes(size=pop)+geom_point(alpha=0.5)
```

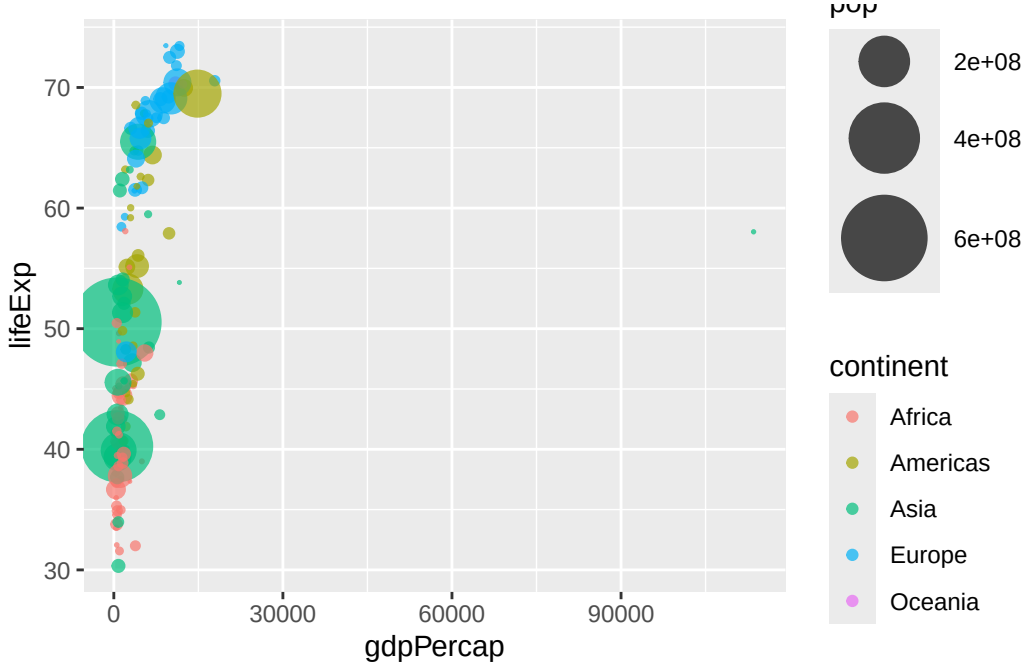


Q3. Can you adapt the code you have learned thus far to reproduce our gapminder scatter plot for the year 1957? What do you notice about this plot is it easy to compare with the one for 2007?

Steps to produce your 1957 plot should include:

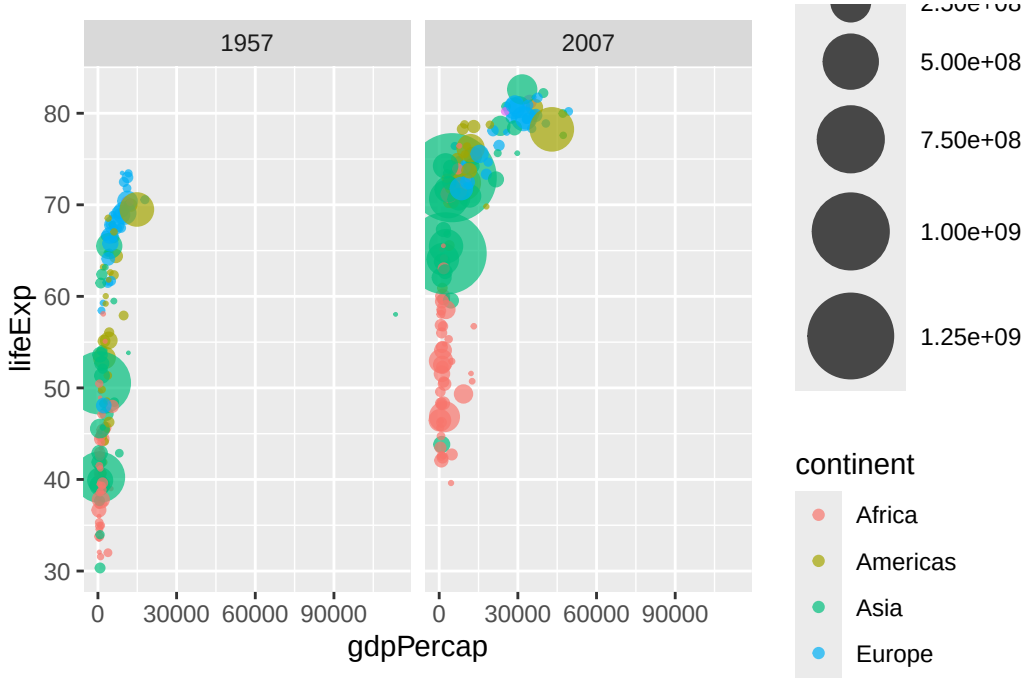
- Use dplyr to `filter` the `gapmider` dataset to include only the year 1957 (check above for how we did this for 2007).
- Save your result as `gapminder_1957`.
- Use the `ggplot()` function and specify the `gapminder_1957` dataset as input
- Add a `geom_point()` layer to the plot and create a scatter plot showing the GDP per capita `gdpPercap` on the x-axis and the life expectancy `lifeExp` on the y-axis
- Use the `color` aesthetic to indicate each `continent` by a different color
- Use the `size` aesthetic to adjust the point size by the population pop
- Use `scale_size_area()` so that the point sizes reflect the actual population differences and set the `max_size` of each point to 15 -Set the opacity/transparency of each point to 70% using the `alpha=0.7` parameter

```
gapminder_1957<-gapminder%>%filter(year==1957)

ggplot(gapminder_1957)+aes(x=gdpPercap, y=lifeExp, color=continent,size = pop) +
  geom_point(alpha=0.7) +
  scale_size_area(max_size = 15)
```



Q4. Do the same steps above but include 1957 and 2007 in your input dataset for `ggplot()`. You should now include the layer `facet_wrap(~year)` to produce the following plot:

```
gapminder_1957<-gapminder%>%filter(year==1957|year==2007)

ggplot(gapminder_1957)+aes(x=gdpPercap, y=lifeExp, color=continent,size = pop) +
  geom_point(alpha=0.7) +
  scale_size_area(max_size = 15)+
  facet_wrap(~year)
```



Combining plots

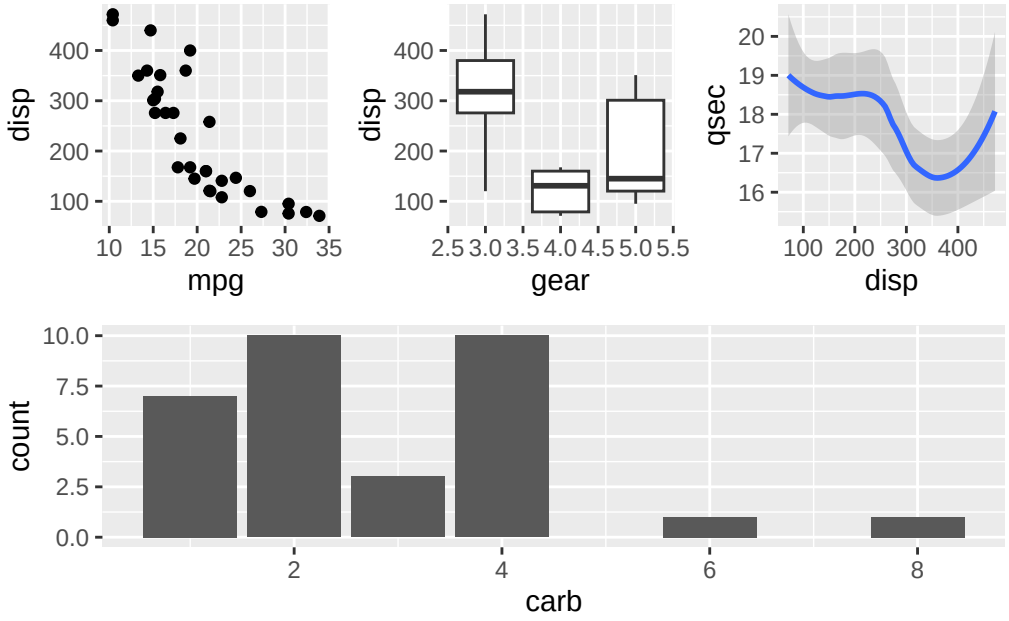
Example:

```
#install.packages("patchwork")
library(patchwork)
```

```
Warning: package 'patchwork' was built under R version 4.3.3
```

```
# Setup some example plots
p1 <- ggplot(mtcars) + geom_point(aes(mpg, disp))
p2 <- ggplot(mtcars) + geom_boxplot(aes(gear, disp, group = gear))
p3 <- ggplot(mtcars) + geom_smooth(aes(disp, qsec))
p4 <- ggplot(mtcars) + geom_bar(aes(carb))

# Use patchwork to combine them here:
(p1 | p2 | p3) /
      p4
```

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


disp
400
300
200
100
10 15 20 25 30 35
mpg
disp
400
300
200
100
2.5 3.0 3.5 4.0 4.5 5.0 5.5
gear
qsec
20
19
18
17
16
100 200 300 400
disp
count
10.0
7.5
5.0
2.5
0.0
2 4 6 8
carb