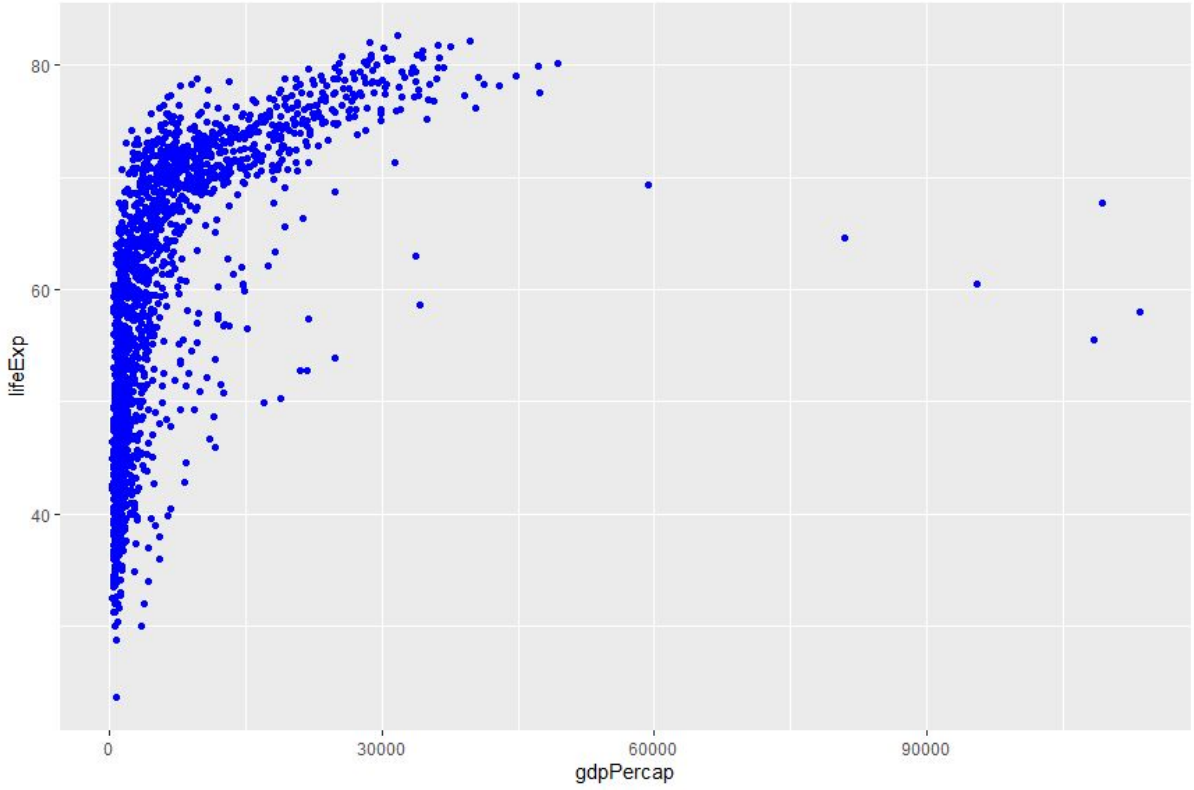
Team 5
Yifan Feng: 2671027
Yunxiang Li: 2674844

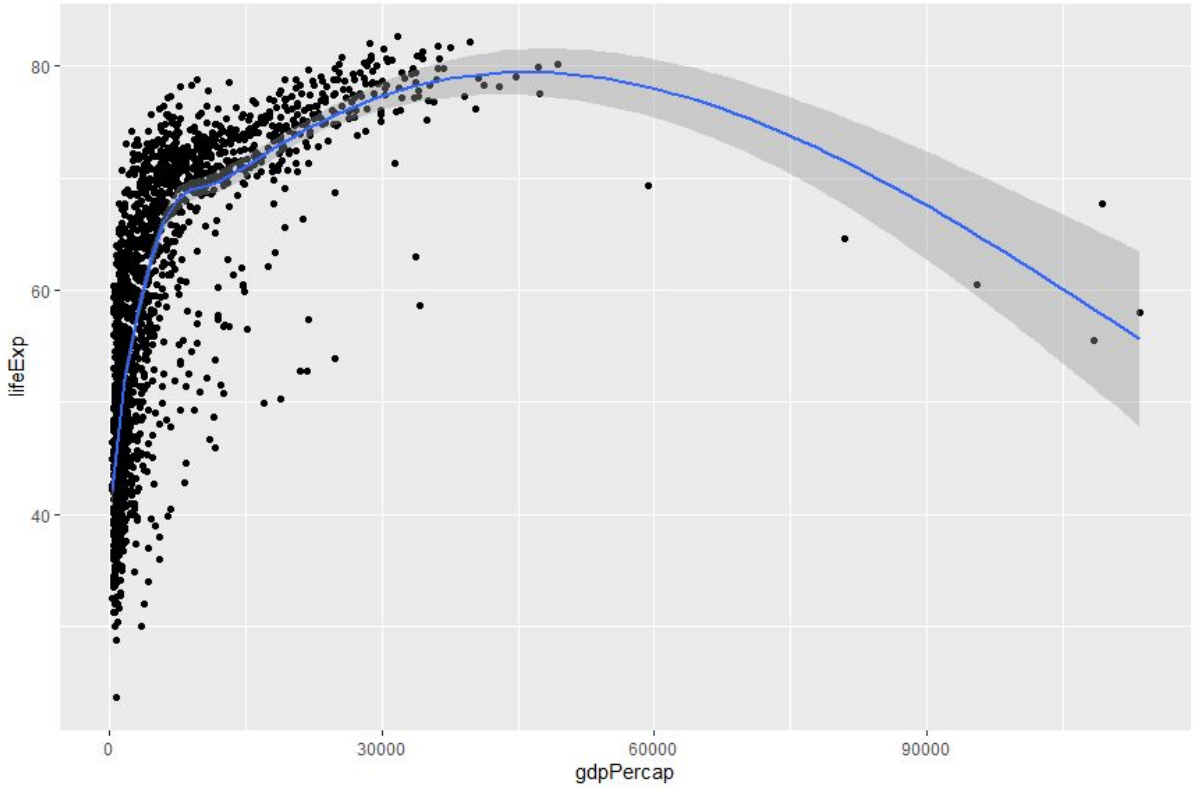
1a)

#The value of the variable color should not be assigned inside the aesthetics statement. By indicating colors inside aes( ), one in fact creates a new variable for the plot with red color as default value.
> ggplot(data = gapminder) + geom_point(mapping = aes(x = gdpPercap, y = lifeExp),color = "blue")



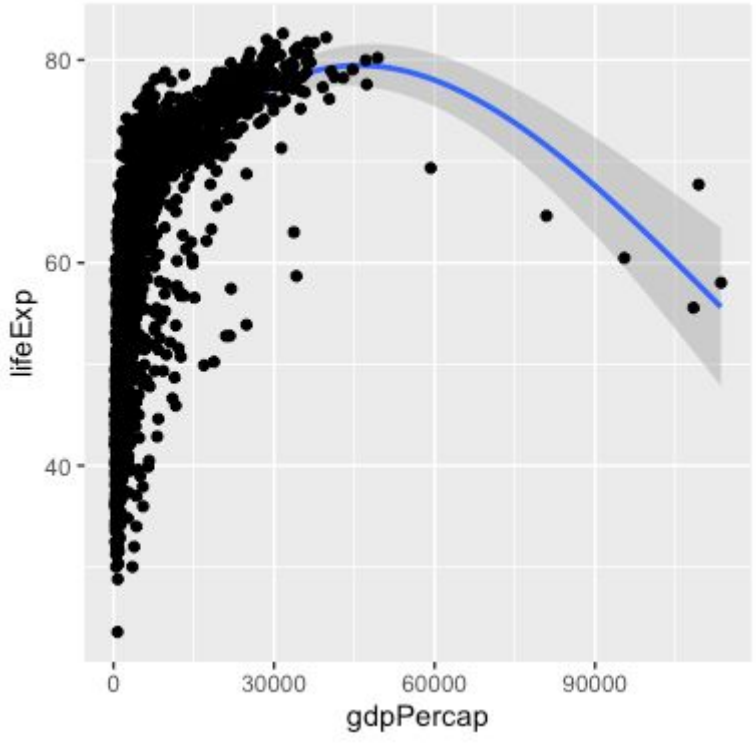
1b)

```
> p <- ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp))
> p + geom_jitter() + geom_smooth()
```

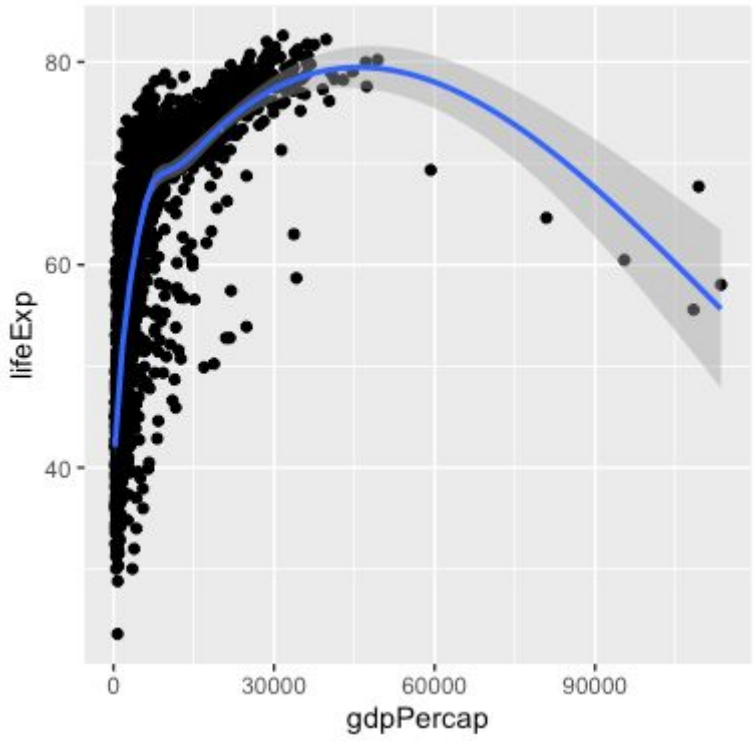


1c)

```
> p + geom_smooth() + geom_point()
```



```
> p + geom_point() + geom_smooth()
```

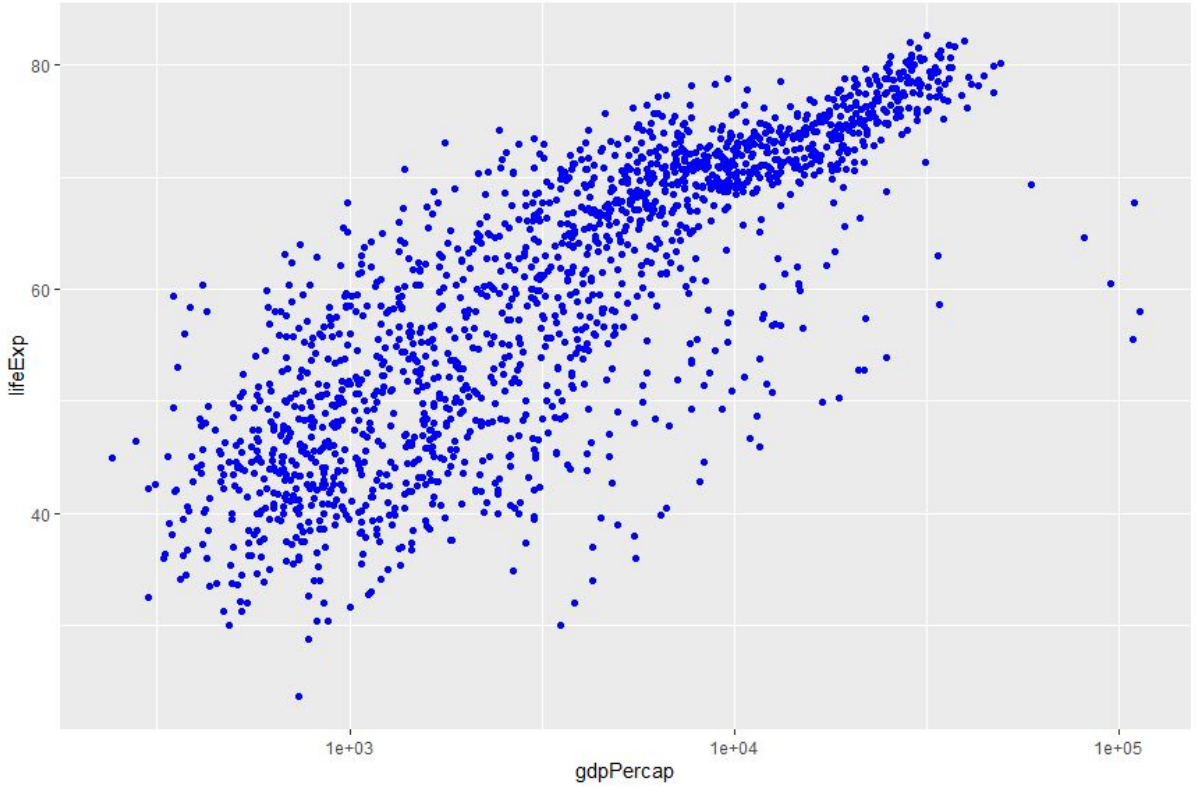



```
#As shown above, the order of geom_point( ) and geom_smooth( ) in codes would affect the order of drawing layers of graphs, meaning that if we call point in the second place, dots will overlap the curve.

#This tells that when a function is called it will be printed immediately, meaning that all codes function in order (from the very left to the very right of a line).

#It could be useful to indicate what kind of results one is expecting. One can stress plot graph by drawing trends or a distribution map of scattered points.
```

1d)

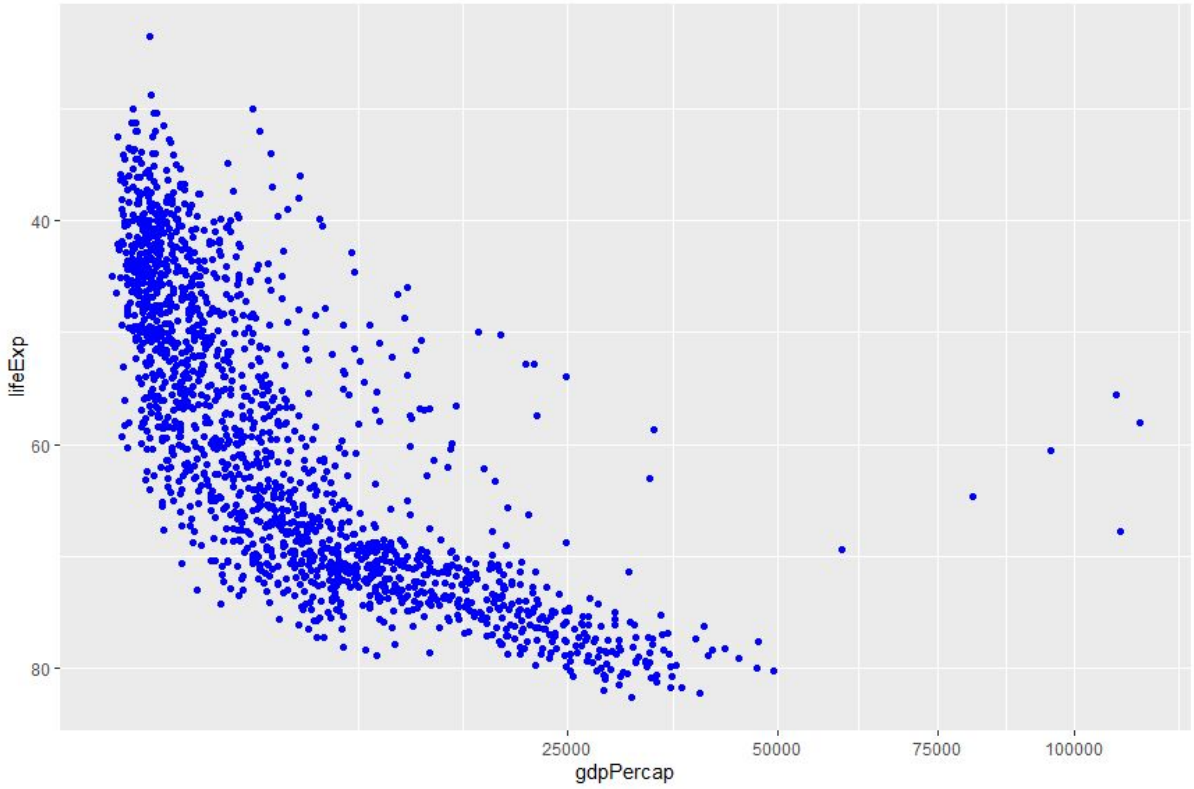
```
# scale_x_log10() changes the scale of the x-axis. Log scale helps to redraw distribution and bring dots closer in the centre.
```

```
> ggplot(data = gapminder) + geom_point(mapping = aes(x = gdpPercap, y = lifeExp),color = "blue") + scale_x_log10()
```



1e)

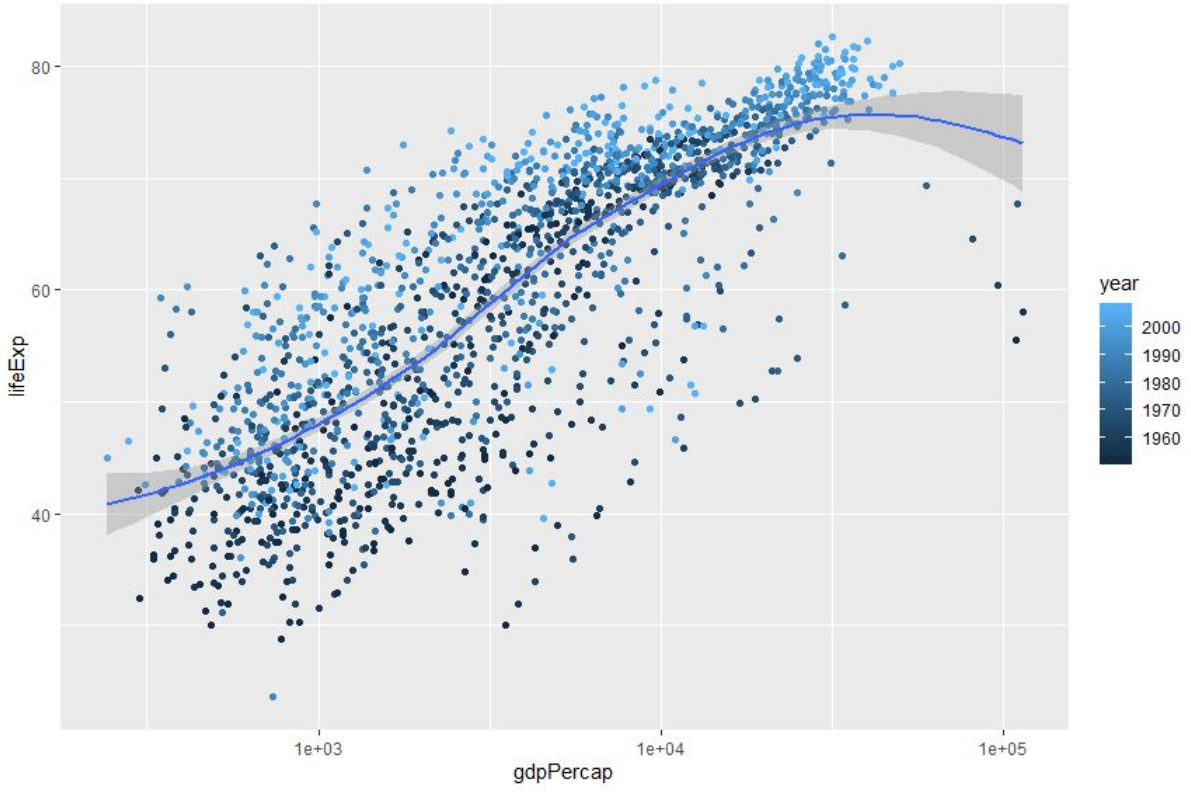
#scale_x_sqrt() produces a value that is a square root of the values on the x-axis. Likewise, scale_y_sqrt() generates the square root of the y-axis values. scale_x_reverse() reverses the scale vertically whilst scale_y_reverse() lays down x-axis (horizontally).
> ggplot(data = gapminder) + geom_point(mapping = aes(x = gdpPercap, y = lifeExp),color = "blue") + scale_x_sqrt() + scale_y_reverse()



1f)

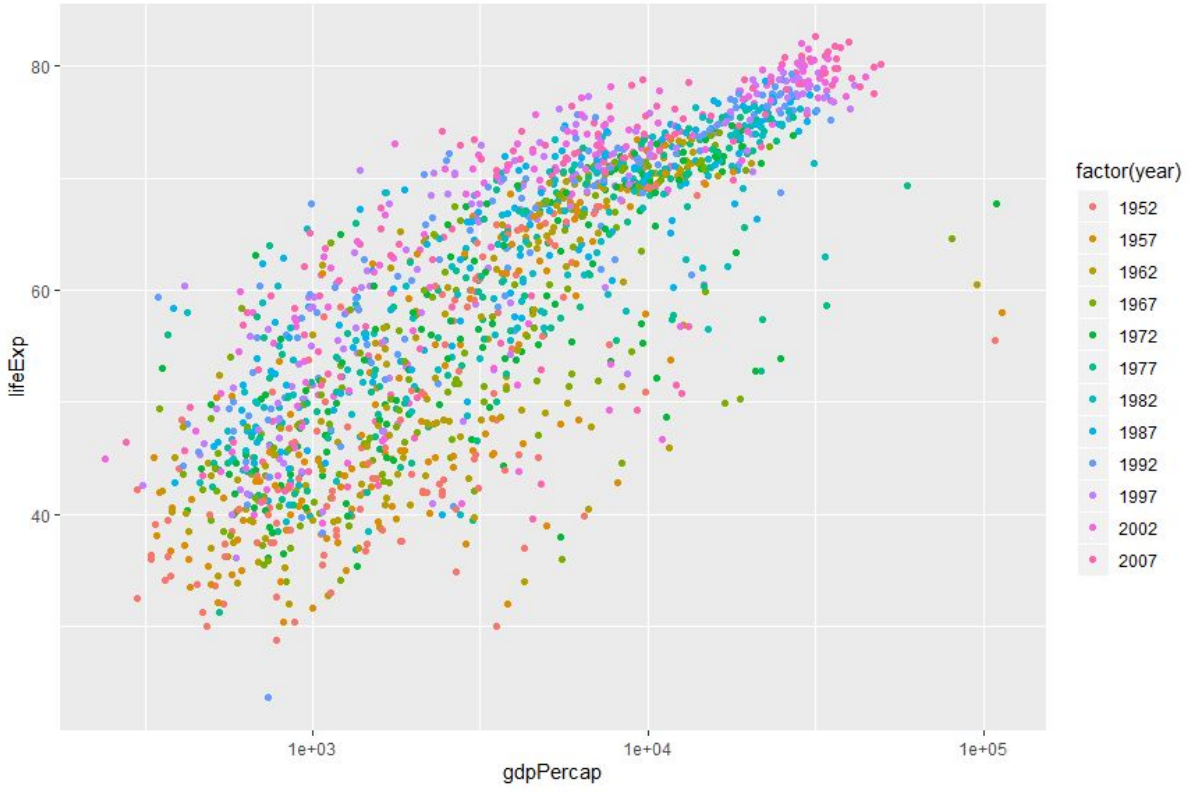
```
#The code gives the different continents different colors
> p <- ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp))
> p + geom_jitter(aes(color = continent)) + geom_smooth(method = "loess") +
scale_x_log10()
```


```
#This line shows the changes along with consecutive years rather than continent.
> p <- ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp))
> p + geom_jitter(aes(color = year)) + geom_smooth(method = "loess") + scale_x_log10()
```



1g)

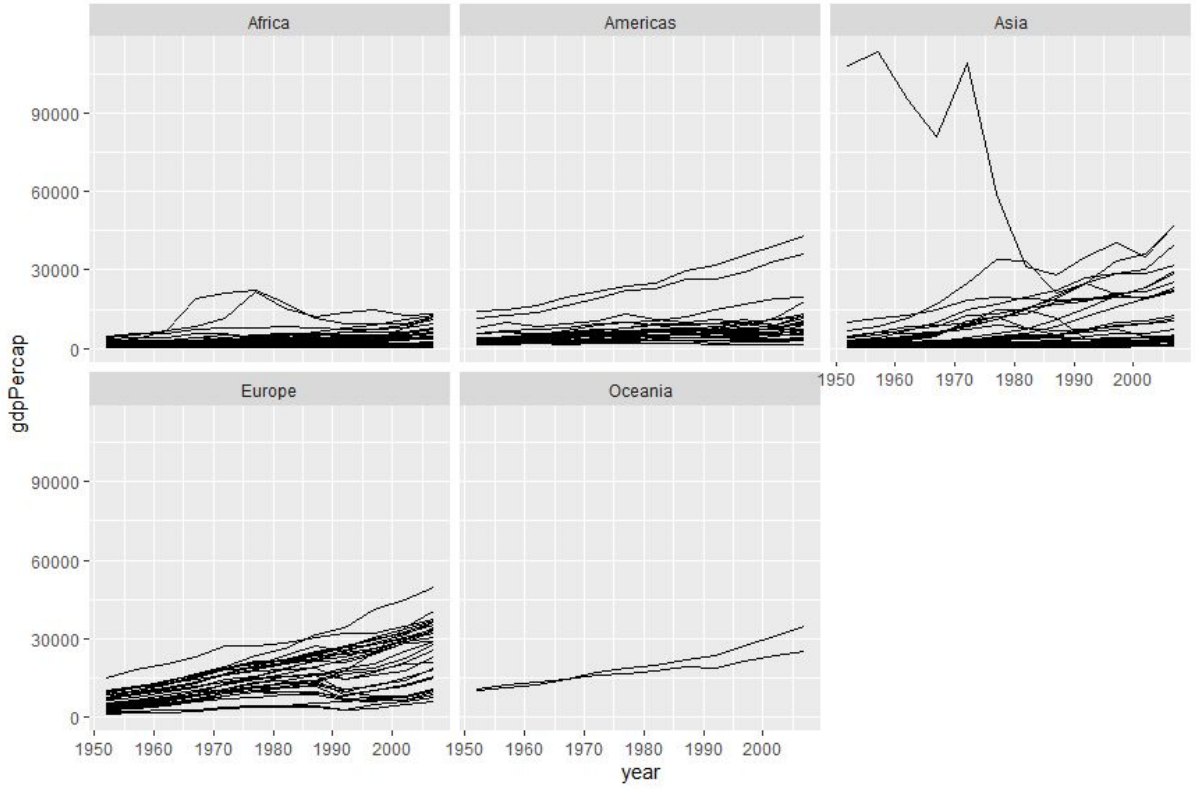
```
#Plots are shown by the individual years with various colors, not with different scales
  > ggplot(data = gapminder) + geom_point(mapping = aes(x = gdpPercap, y = lifeExp,
color = factor(year))) + scale_x_log10()
```



2a)

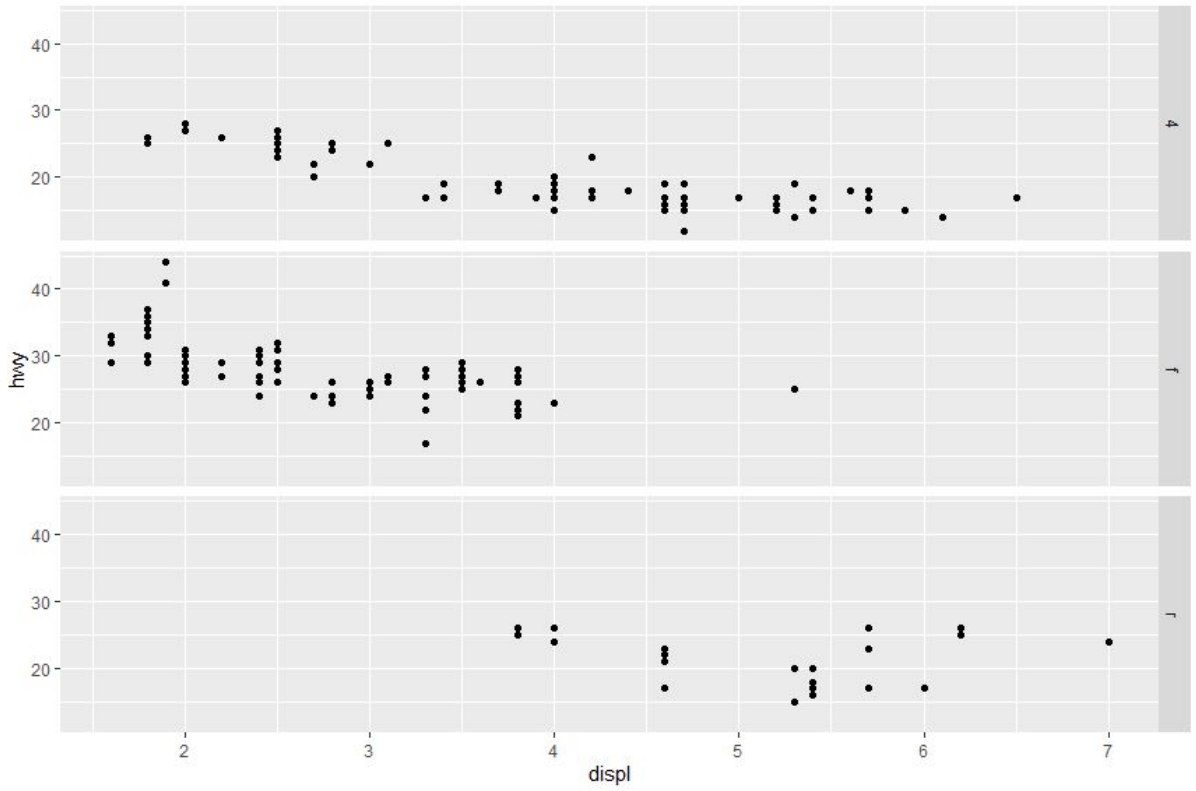
#The original line graph contains all countries gdp trend which is unclear to read. We want to display lines based on individual regions (by continent), so we divide the originals into five facets.

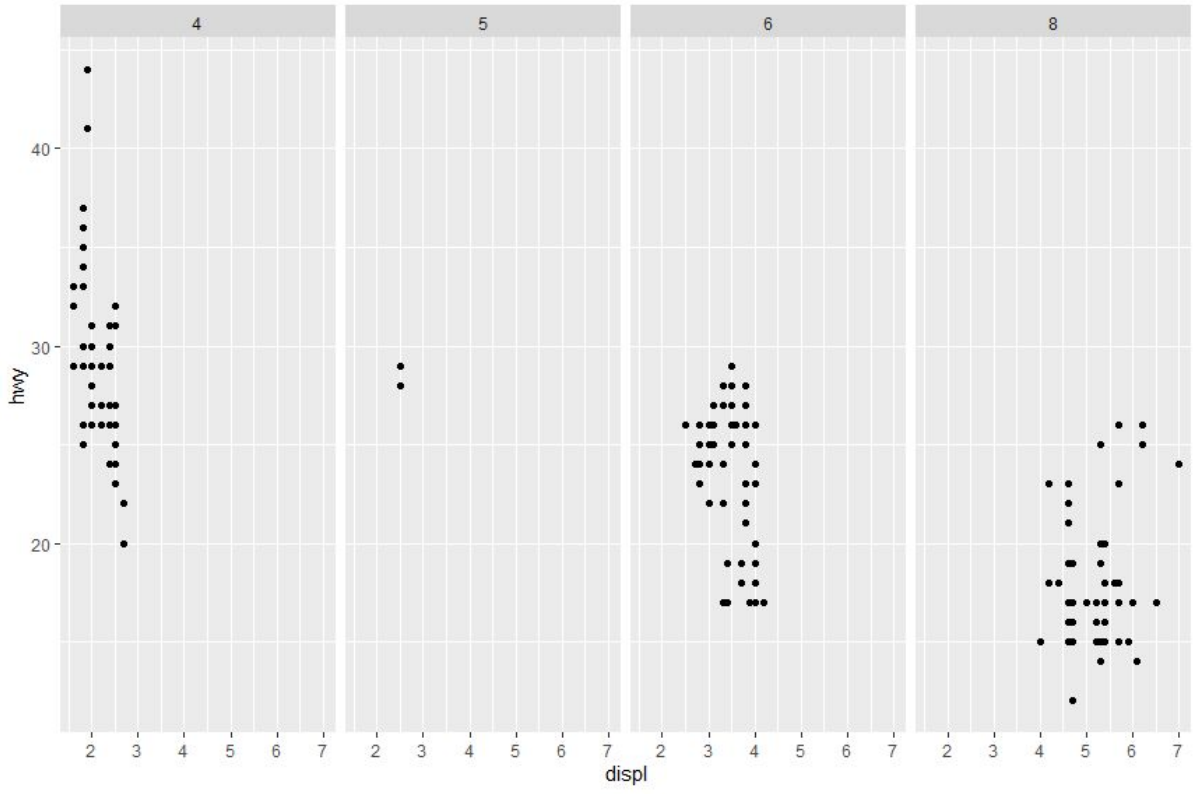
```
> p <- ggplot(data = gapminder, mapping = aes(x = year, y = gdpPercap))
> p + geom_line(aes(group = country)) + facet_wrap(~ continent)
```



2b)

#Graph one uses facet_grid (drv ~.) to draw facet into rows based on drv value; whilst graph two uses (.~cyl) to divide facets into columns based on cyl value. The dot (.) functions as an indicator to show there is no division on the dimension (either row or column).


4
f
r
hwy
40
30
20
2
3
4
5
6
7
displ


4
5
6
8
hwy
40
30
20
2
3
4
5
6
7
displ

2c)

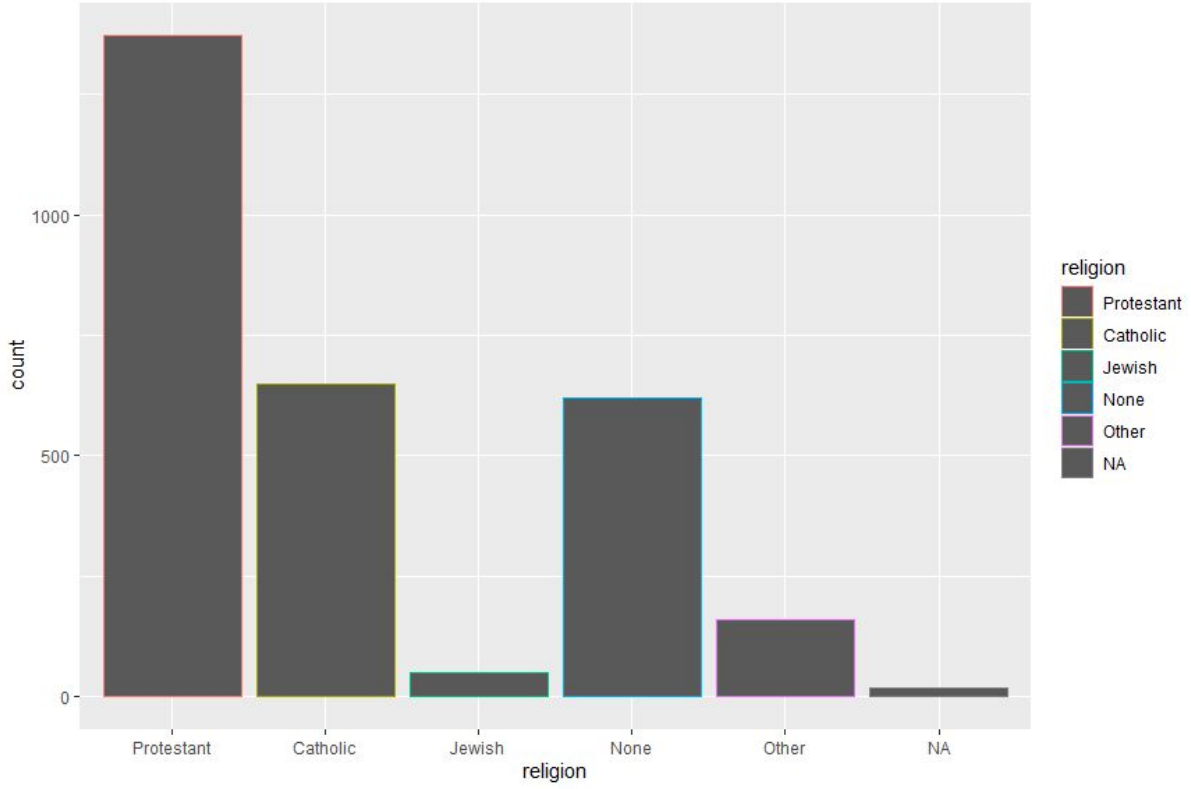
1. nrow: define the number of rows of the panels
2. ncol: define the number of columns of the panels
3. Other options: scales, shrink, labeller, as.table, switch, drop, dir, strip.position
4. Because in facet_grid, the number of unique values of the variables is equal to the number of rows and columns.

2d)

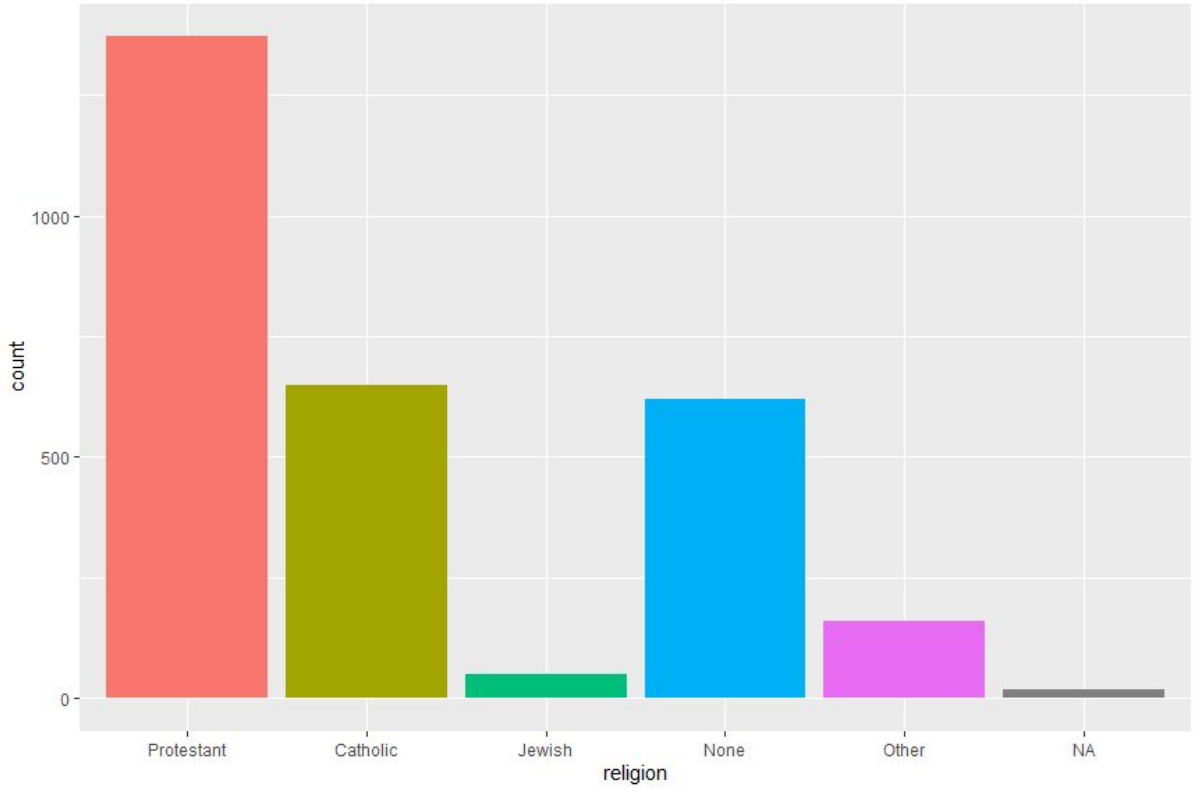
#If putting too much levels on the x-axis, the y-axis will shrink and therefore it will be harder to see values in the plot. It is much easier to compare the values and the trends of the variables if we put the variable with more unique levels in the columns.

3a)

```
> p <- ggplot(data = gss_sm, mapping = aes(x = religion, color = religion))
> p + geom_bar()
#The parameter color changes the outline color of the bar charts.
```

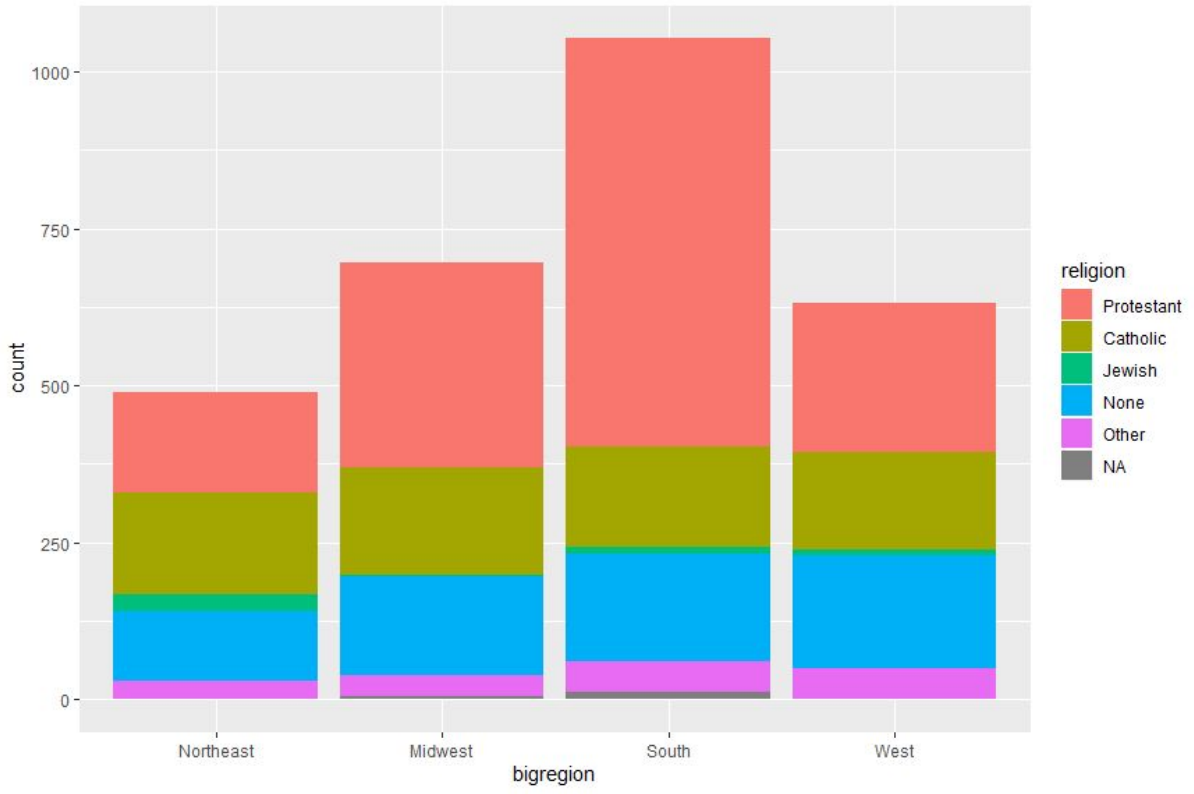


```
> p <- ggplot(data = gss_sm, mapping = aes(x = religion, fill = religion))
> p + geom_bar() + guides(fill = FALSE)
#The parameter fill changes the filling color of the bar charts.
```



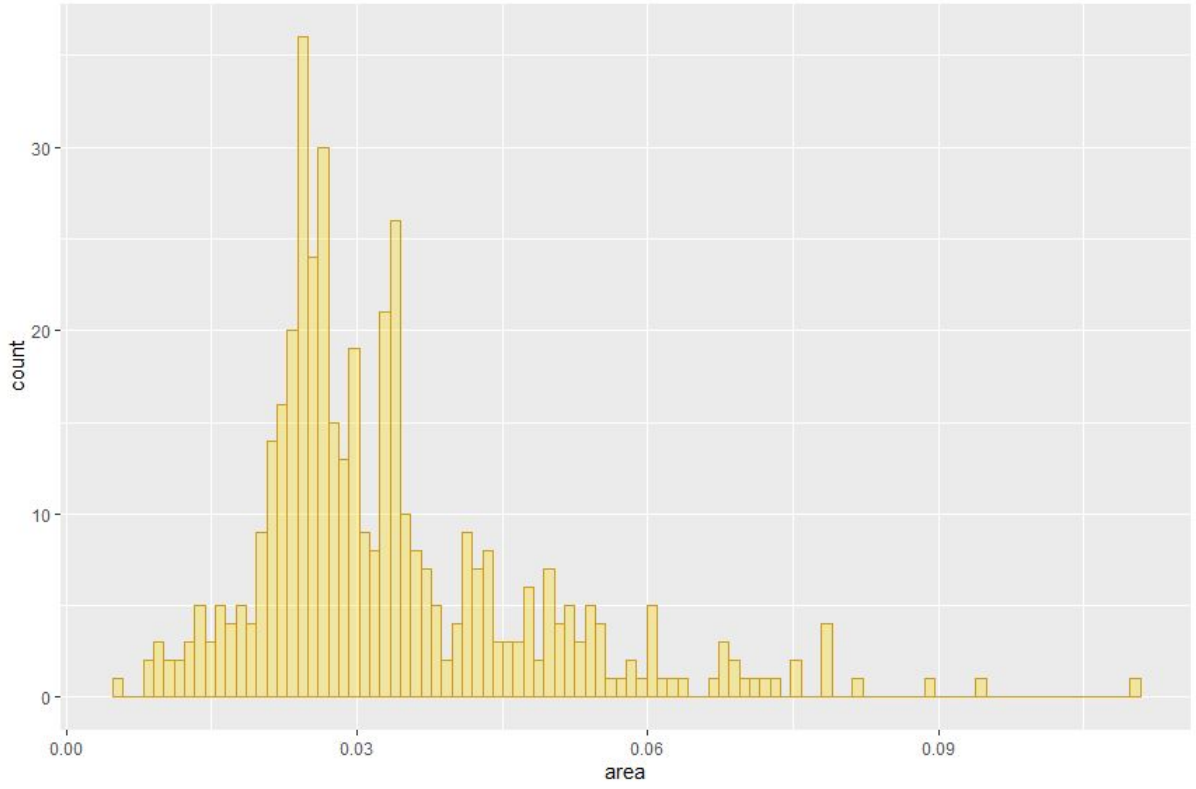
3b)

```
> p <- ggplot(data = gss_sm, mapping = aes(x = bigregion, fill = religion))
> p + geom_bar(mapping = aes (y = ..count..))
```



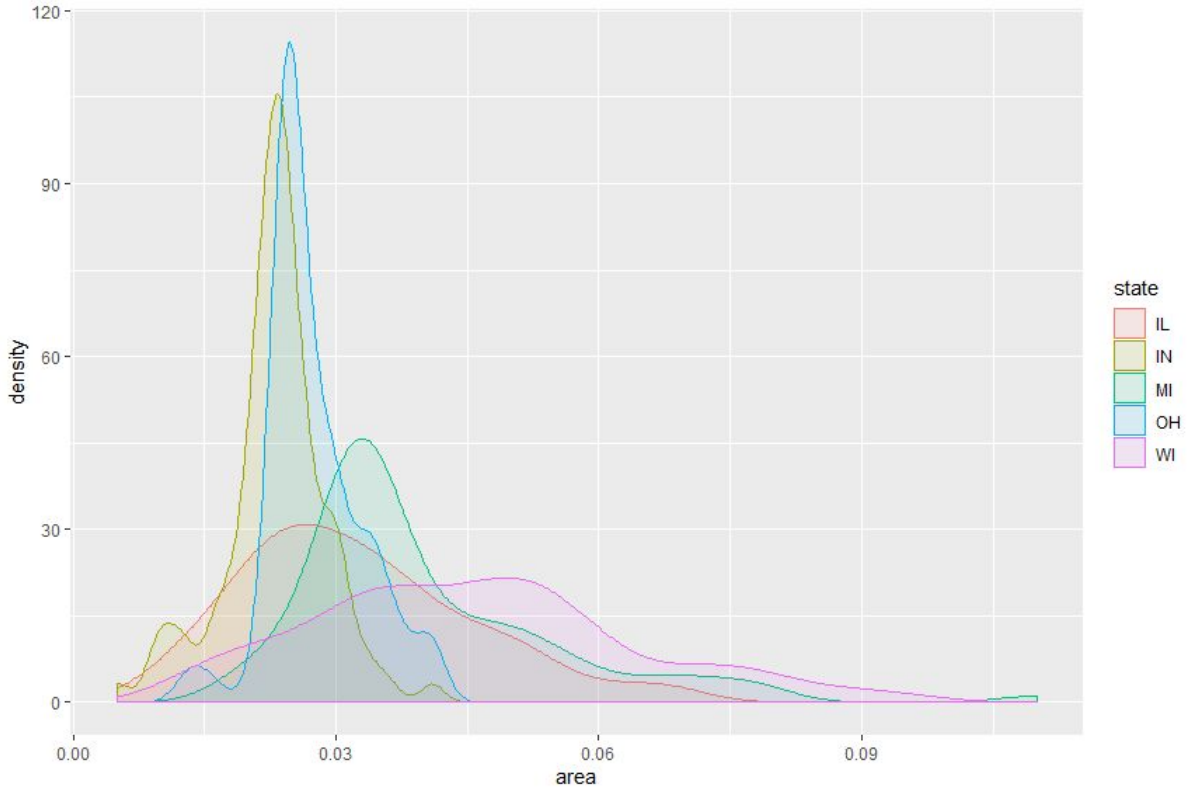
3c)

```
> p <- ggplot(data = midwest, mapping = aes(x = area))
> p + geom_histogram(bins = 100, fill = "gold",color = "goldenrod3",size = 0.5, alpha = 0.3)
```



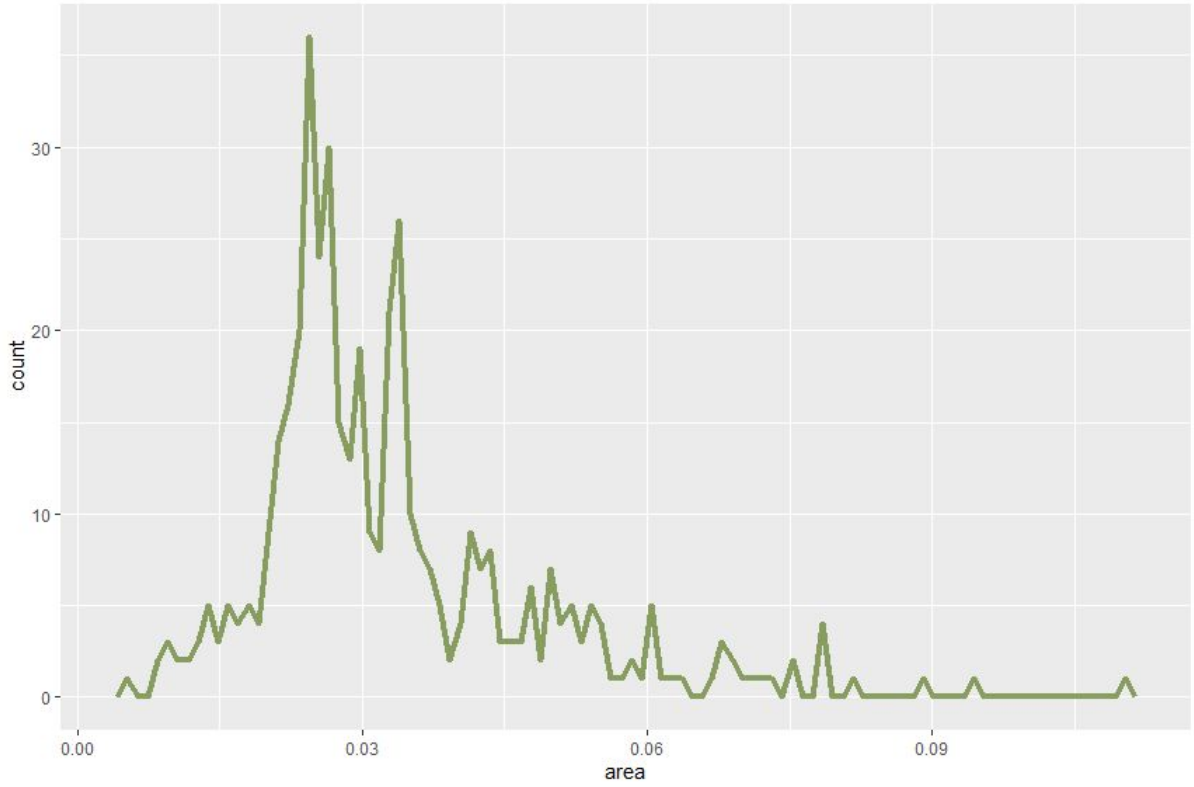
3d)

```
> p <- ggplot(data = midwest, mapping = aes(x = area, fill = state, color = state))
> p + geom_density(alpha = 0.1)
```



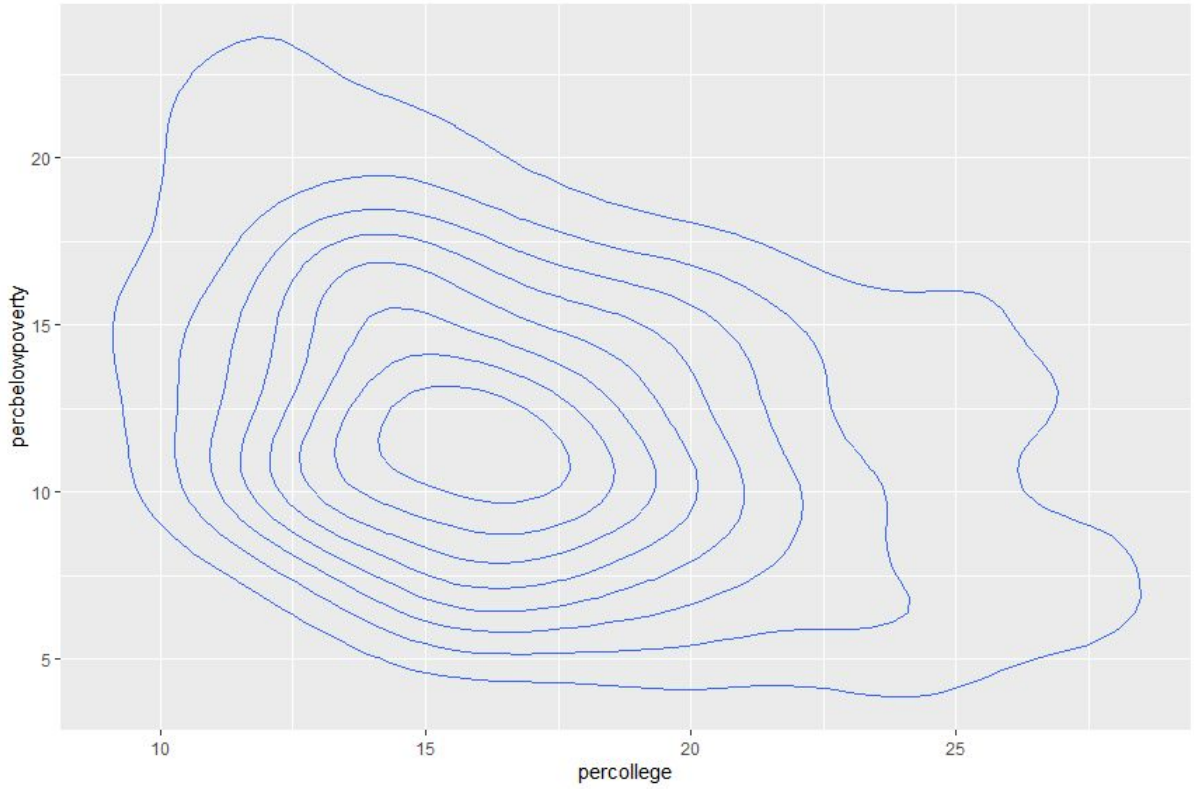
3e)

```
> p <- ggplot(data = midwest, mapping = aes(x = area))
> p + geom_freqpoly(bins = 100, color = "darkolivegreen4", size = 1.5, alpha = 0.8)
```

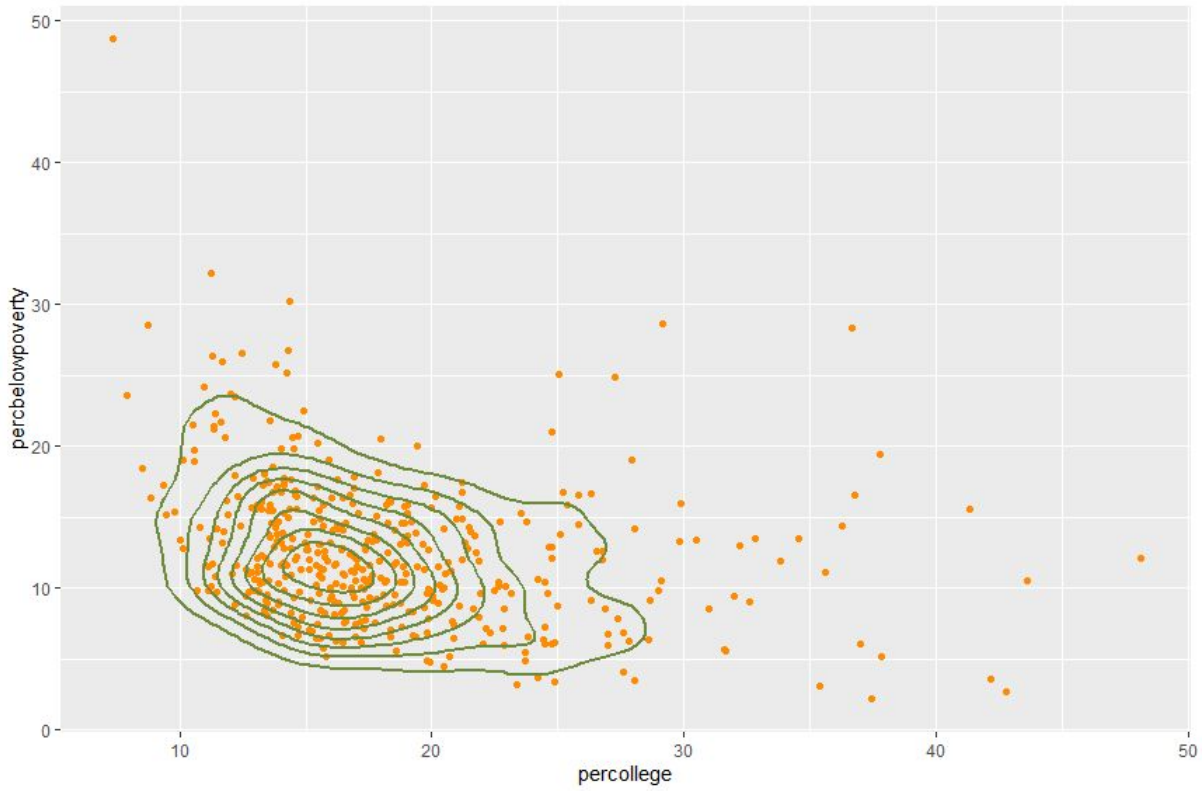


3f)

```
#without geom_point():
> p <- ggplot(data = midwest, mapping = aes(x = percbelowpoverty, y = percollege))
> p + geom_density2d()
```




```
#with geom_point():
> p <- ggplot(data = midwest, mapping = aes(x = percbelowpoverty, y = percollege))
> p + geom_point(color = "darkorange") + geom_density2d(color = "darkolivegreen4",size = 1)
```


50
40
30
20
10
0
percbelowpoverty
10
20
30
40
50
percollege