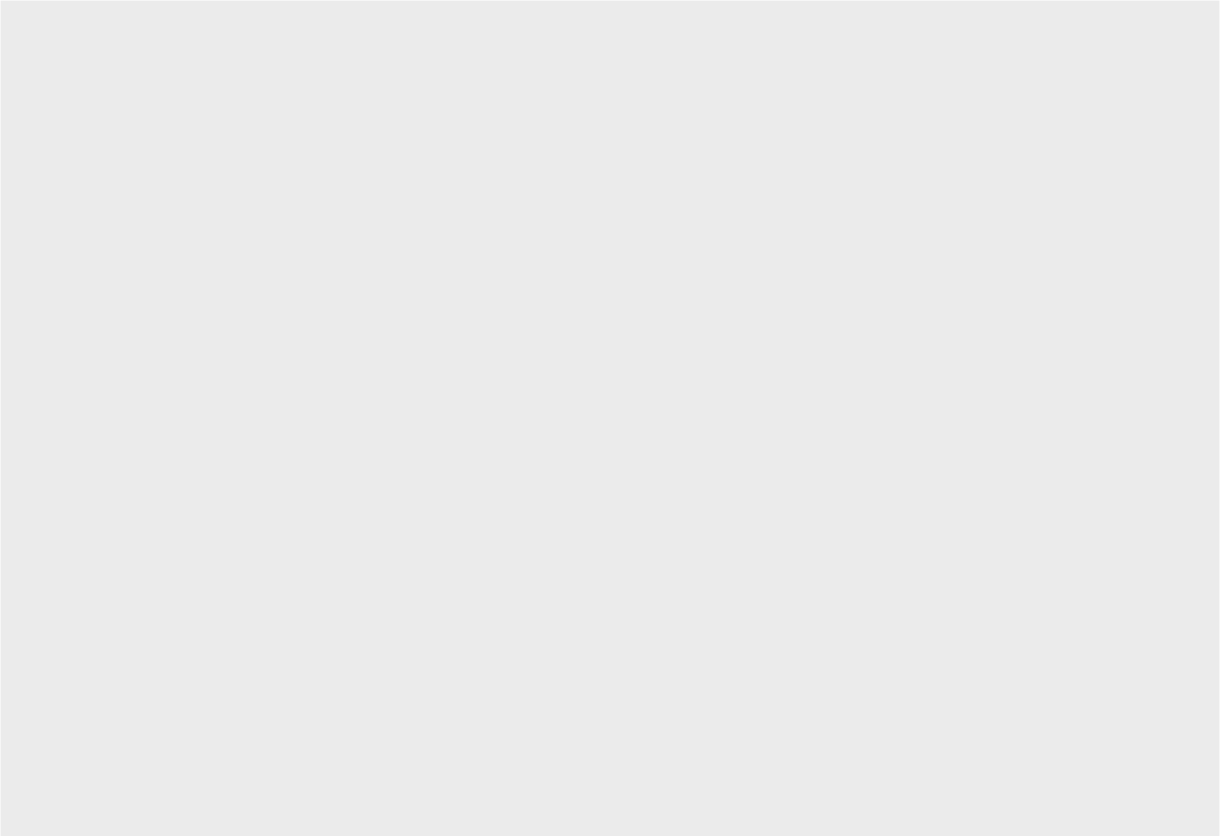
Lab 5: Data Visualization with ggplot2

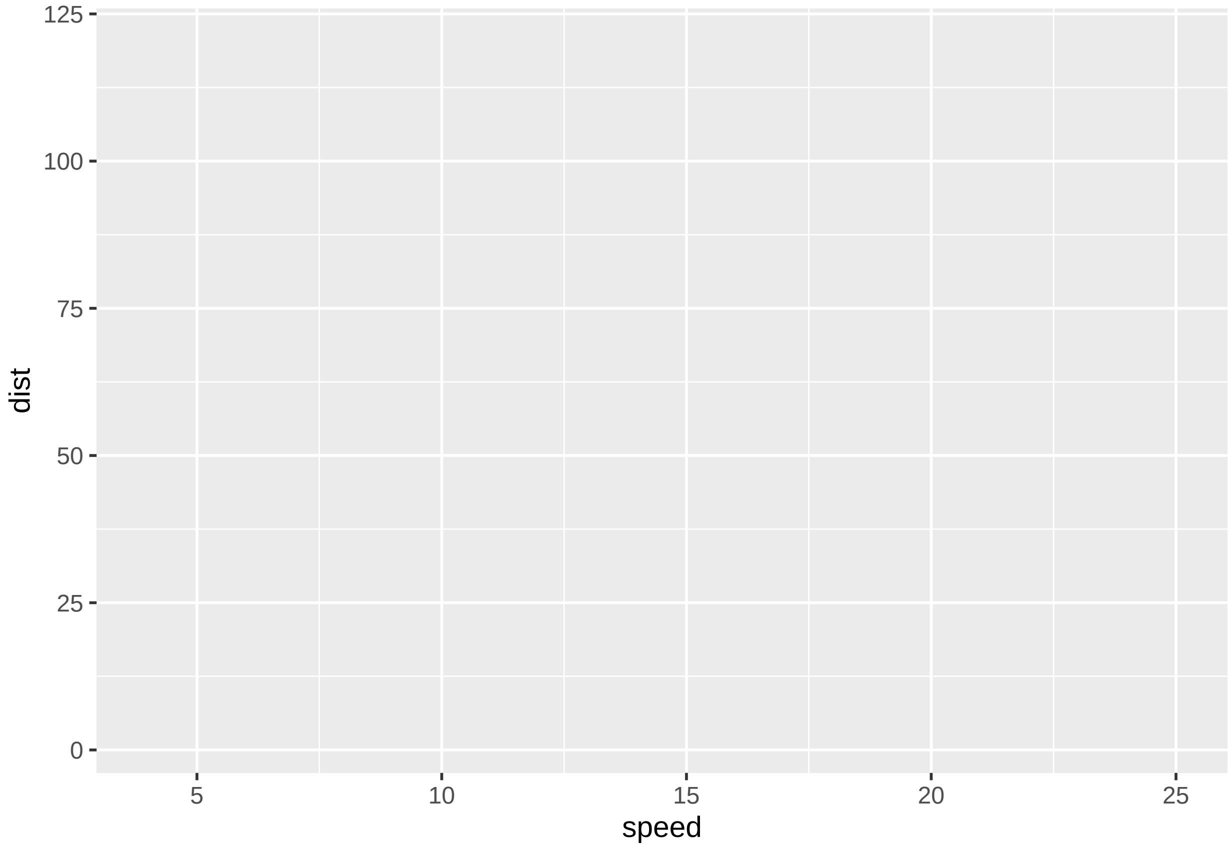
Yushi Li (A15639705)

Feb 5th, 2022

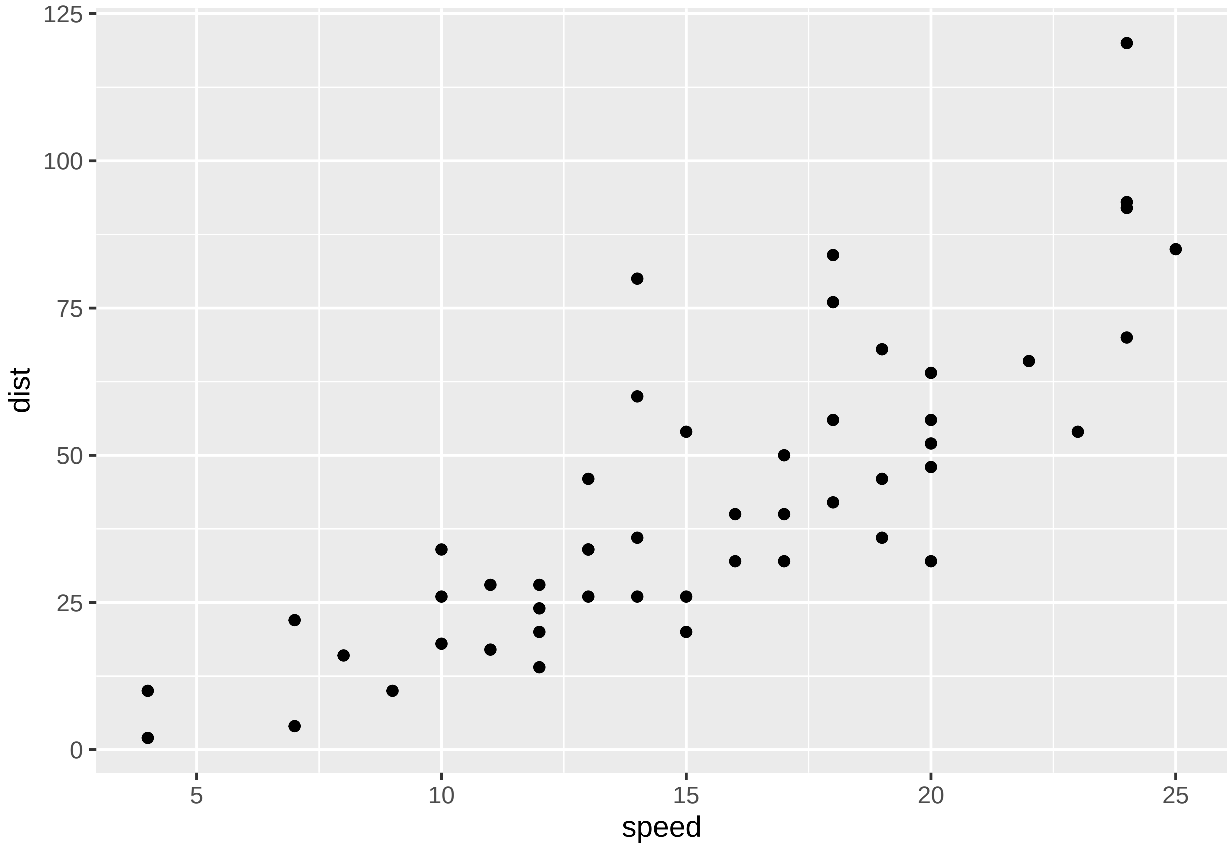
```r
# Lab 5 Data Visualization


# Q.1 For which phases is data visualization important in our scientific
# workflows
# A. All of the above.


# Q.2 True or False? The ggplot2 package comes already installed with R?
# A. False


# Q.3 Which plot types are typically NOT used to compare distributions of
# numeric variables?
# A. Network graphs


# Q.4 Which statement about data visualization with ggplot2 is incorrect?
# A. ggplot2 is the only way to create plots in R.


# install ggplot2
# install.packages(ggplot2)


# load ggplot2
library(ggplot2)


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# Specifying a data set with ggplot()
ggplot(cars)
```



```r
# Specifing aesthetic mappings with aes()
ggplot(cars) +
  aes(x = speed, y = dist)
```

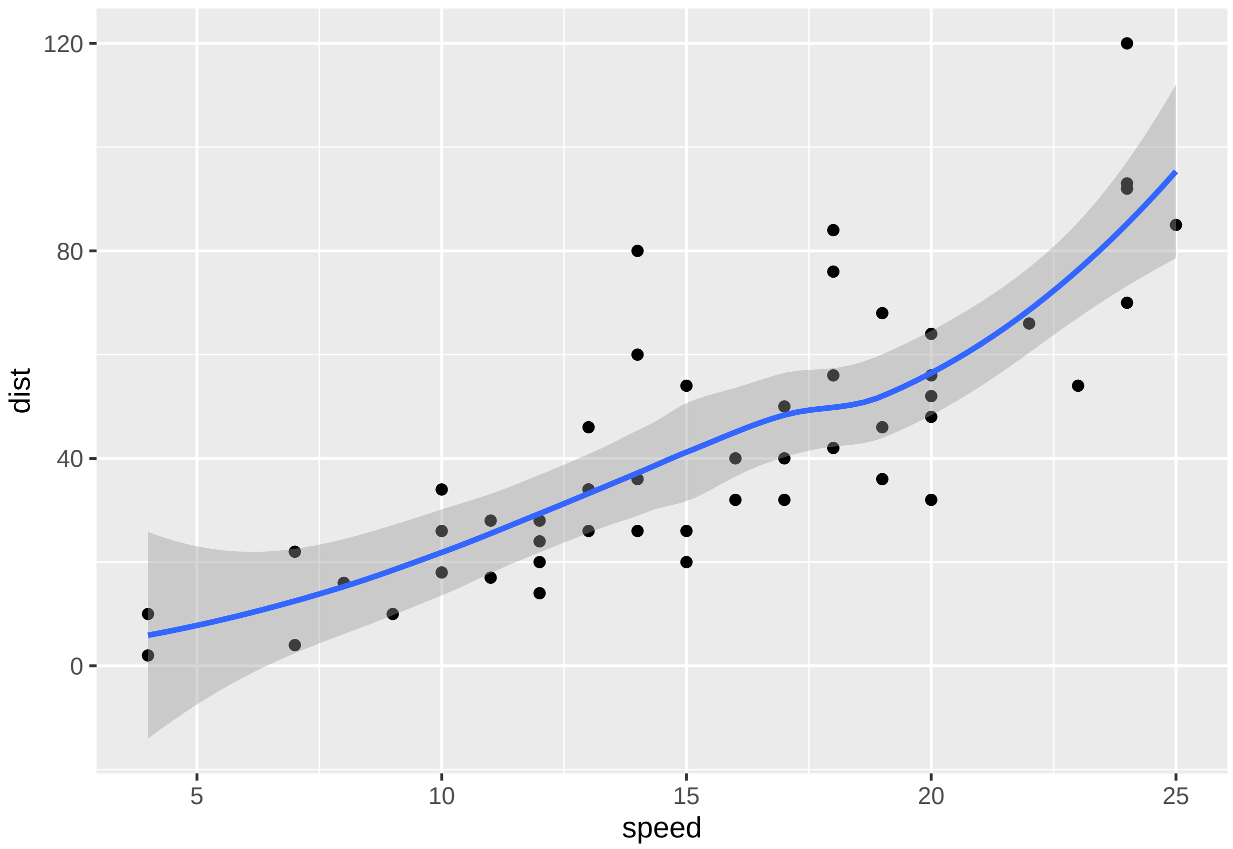


```
# Specifing a geom layer with geom_point()
ggplot(cars) +
  aes(x = speed, y = dist) +
  geom_point()
```



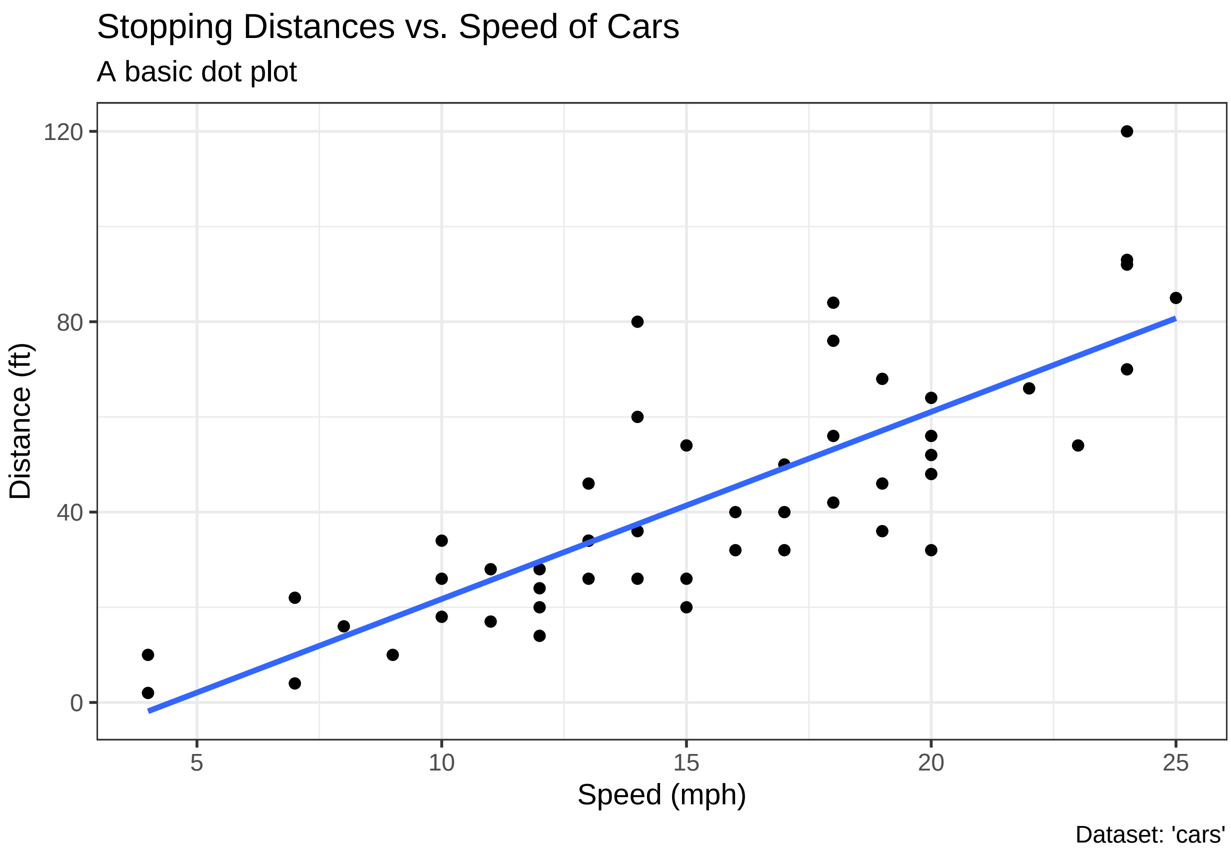
```
# Q.5 Which geometric layer should be used to create scatter plots in ggplot2?
# A. geom_point()


# Q.6 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(cars) +
  aes(x = speed, y = dist) +
  geom_point() +
  geom_smooth()
```

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



```
# Q. 7 Q. 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(cars) +
  aes(x = speed, y = dist) +
  geom_point() +
  labs(title = "Stopping Distances vs. Speed of Cars",
       x = "Speed (mph)",
       y = "Distance (ft)",
       subtitle = "A basic dot plot",
       caption = "Dataset: 'cars'") +
  geom_smooth(method="lm", se=FALSE) +
  theme_bw()
```

```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# Adding more plot aesthetics through aes()
# Loading genes from URL
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
```

```
# Q.8 Use the nrow() function to find out how many genes are in this dataset.
# What is your answer?
nrow(genes)
```

```
## [1] 5196
```

```
# A. 5196
```

```
# Q.9 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?
colnames(genes)
```

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

```
ncol(genes)
```

```
## [1] 4
```

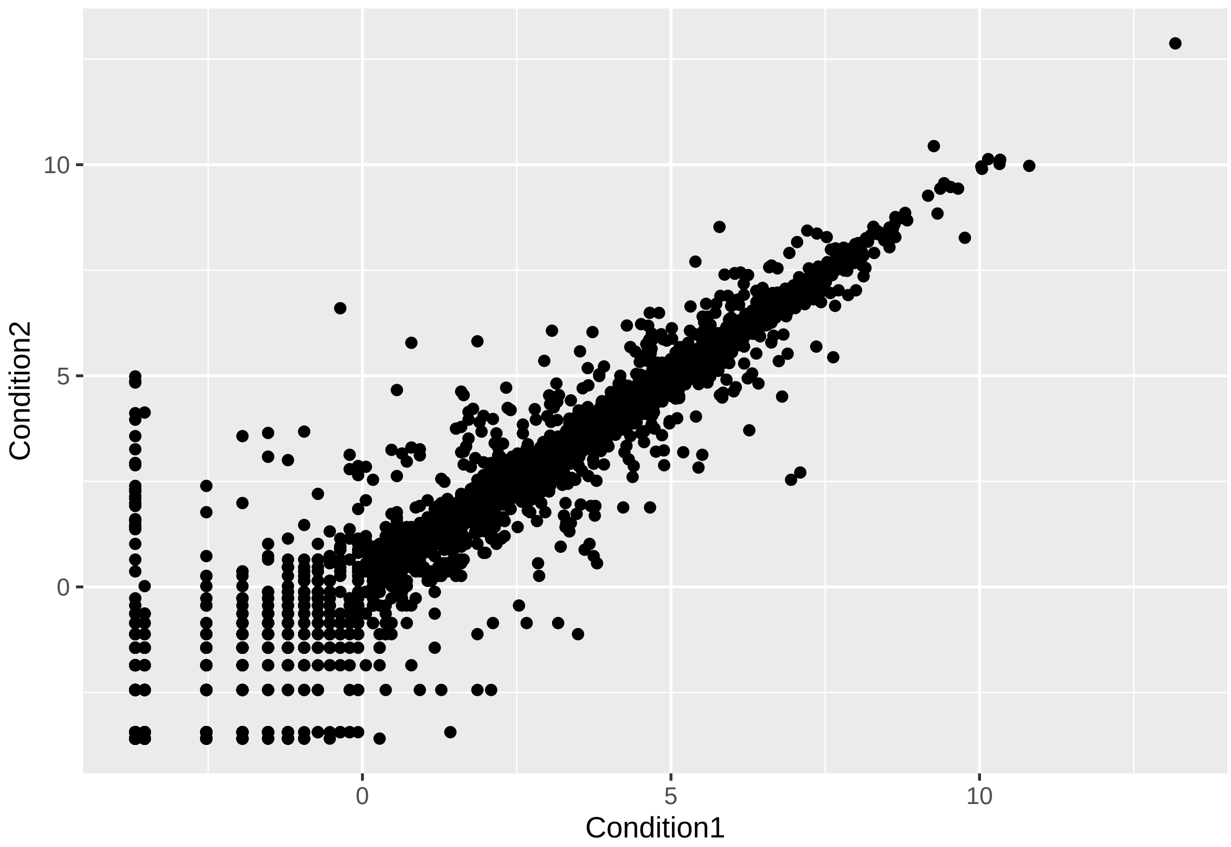
```
# A. 4 columns were found.


# Q.10 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?
table(genes$State)
```

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

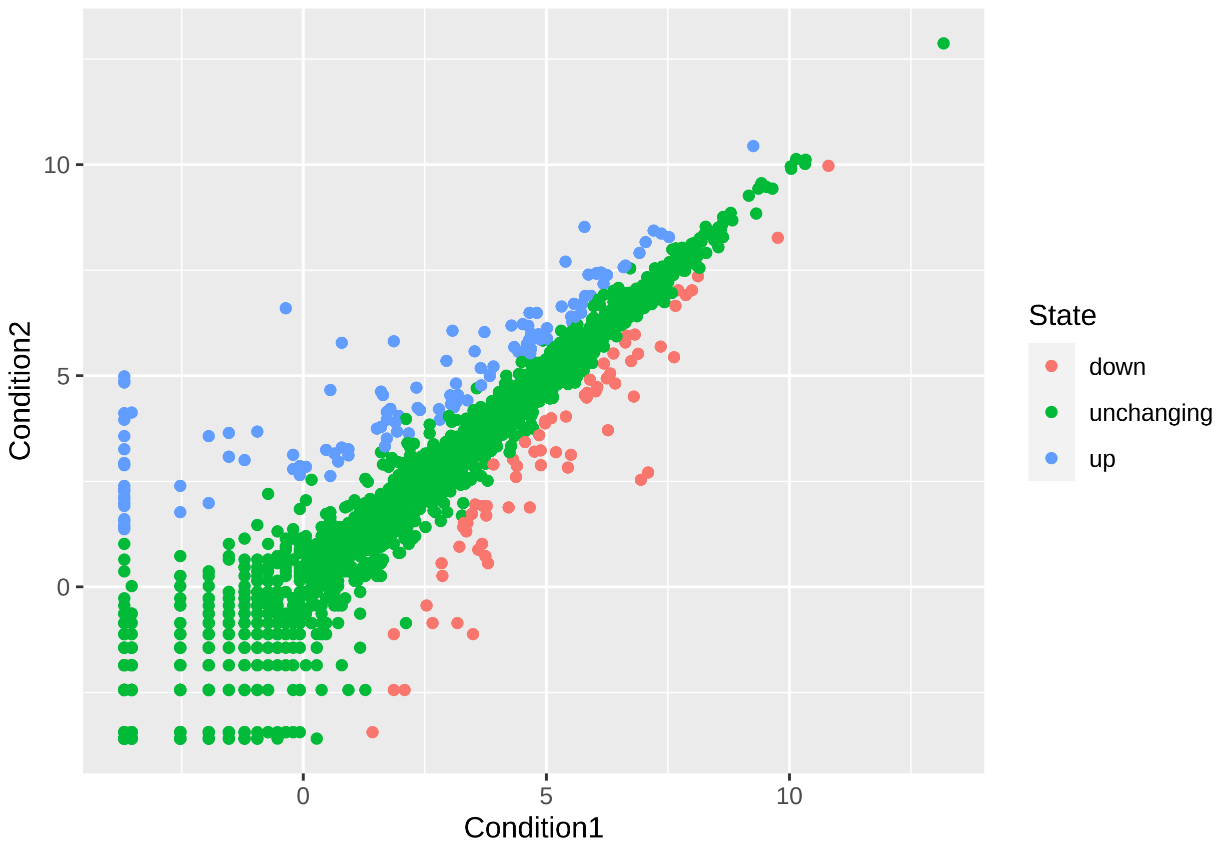
```
# A. 127 genes are up-regulated.


# Q.11 Using your values above and 2 significant figures. What fraction of total
# genes is up-regulated in this dataset?
round(table(genes$State)/nrow(genes)*100, 2)
```

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

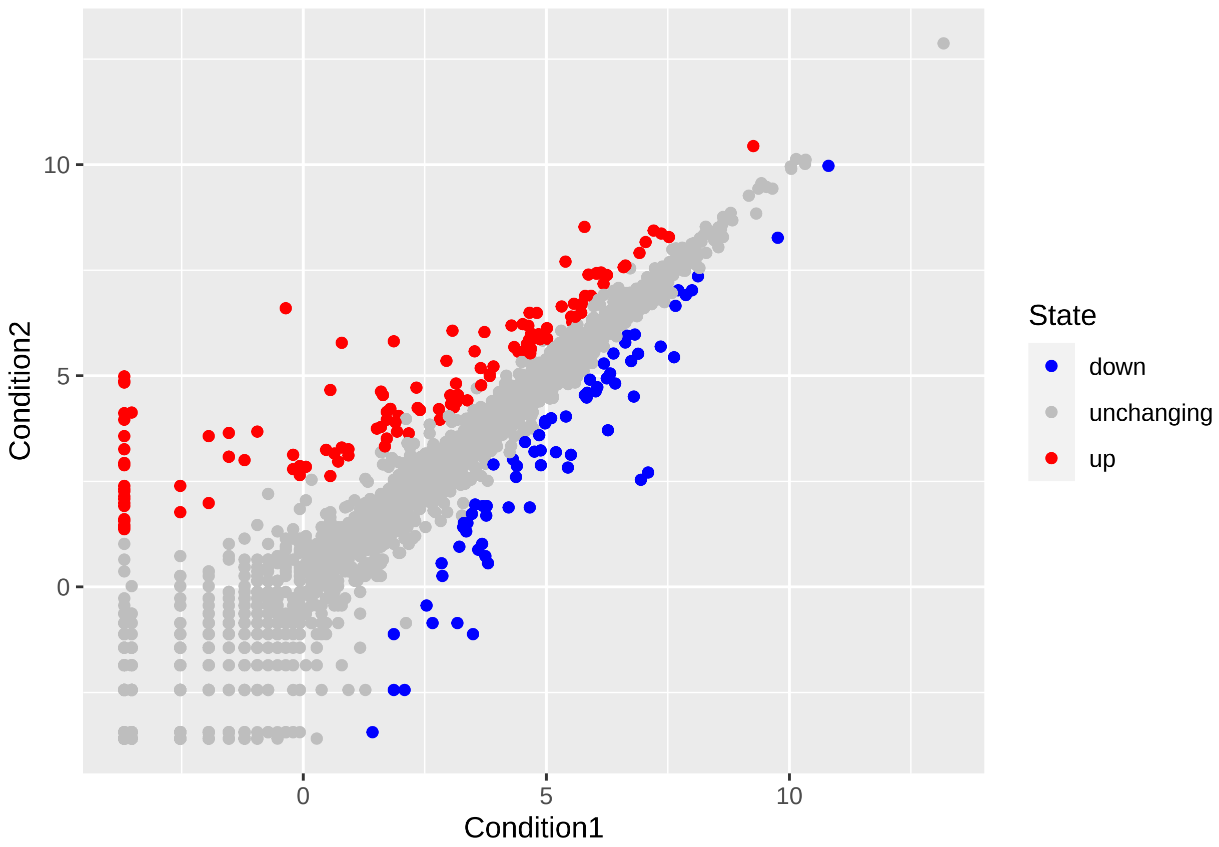
```
# A. 2.44% of total genes are up-regulated.


# Q.12 Complete the code below to produce the following plot:
ggplot(genes) +
  aes(x = Condition1, y = Condition2) +
  geom_point()
```



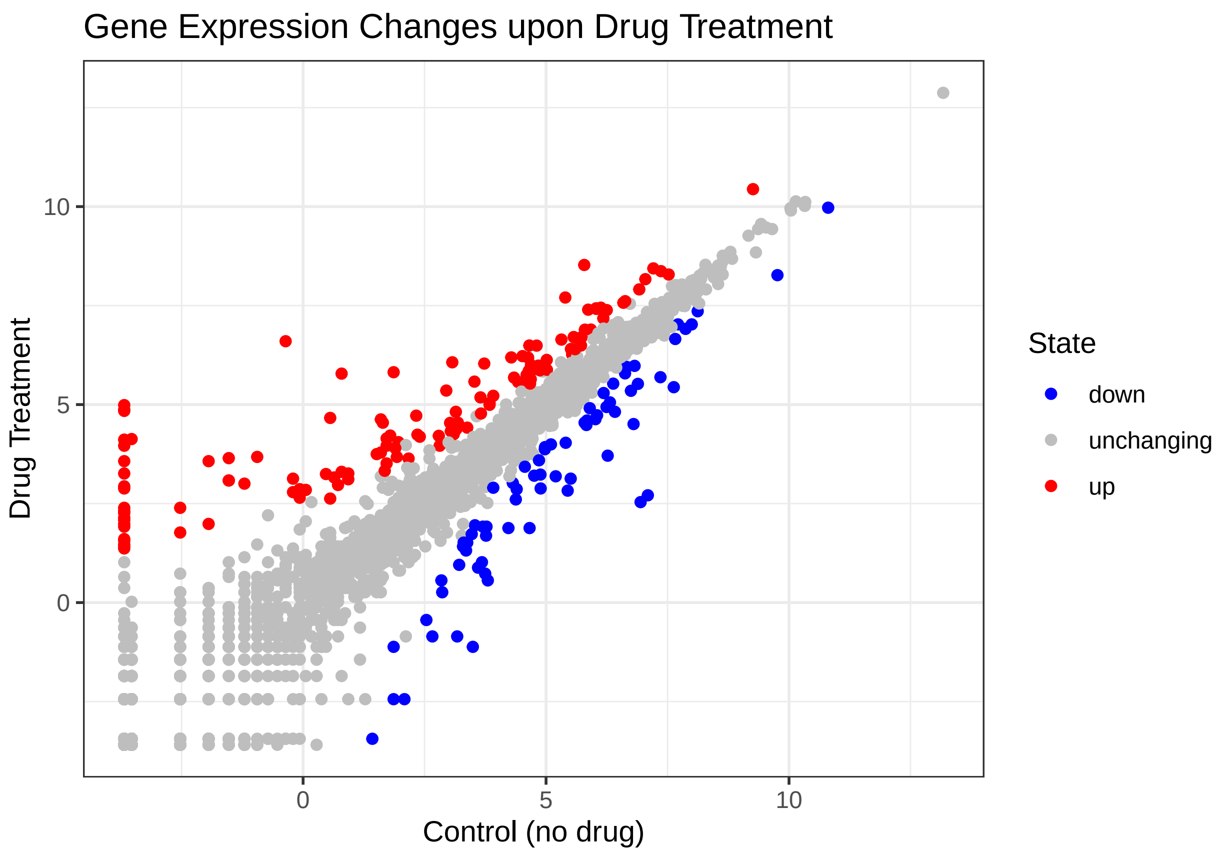
```
# Mapping color to State
p <- ggplot(genes) +
  aes(x = Condition1, y = Condition2, col=State) +
  geom_point()
p
```



```
# Change colors
q <- p + scale_color_manual(values = c("blue", "grey", "red"))
q
```



```
# Q.13 Nice, now add some plot annotations to the p object with the labs()
# function so your plot looks like the following:
q + labs(title = "Gene Expression Changes upon Drug Treatment",
         x = "Control (no drug)",
         y = "Drug Treatment") +
  theme_bw()
```

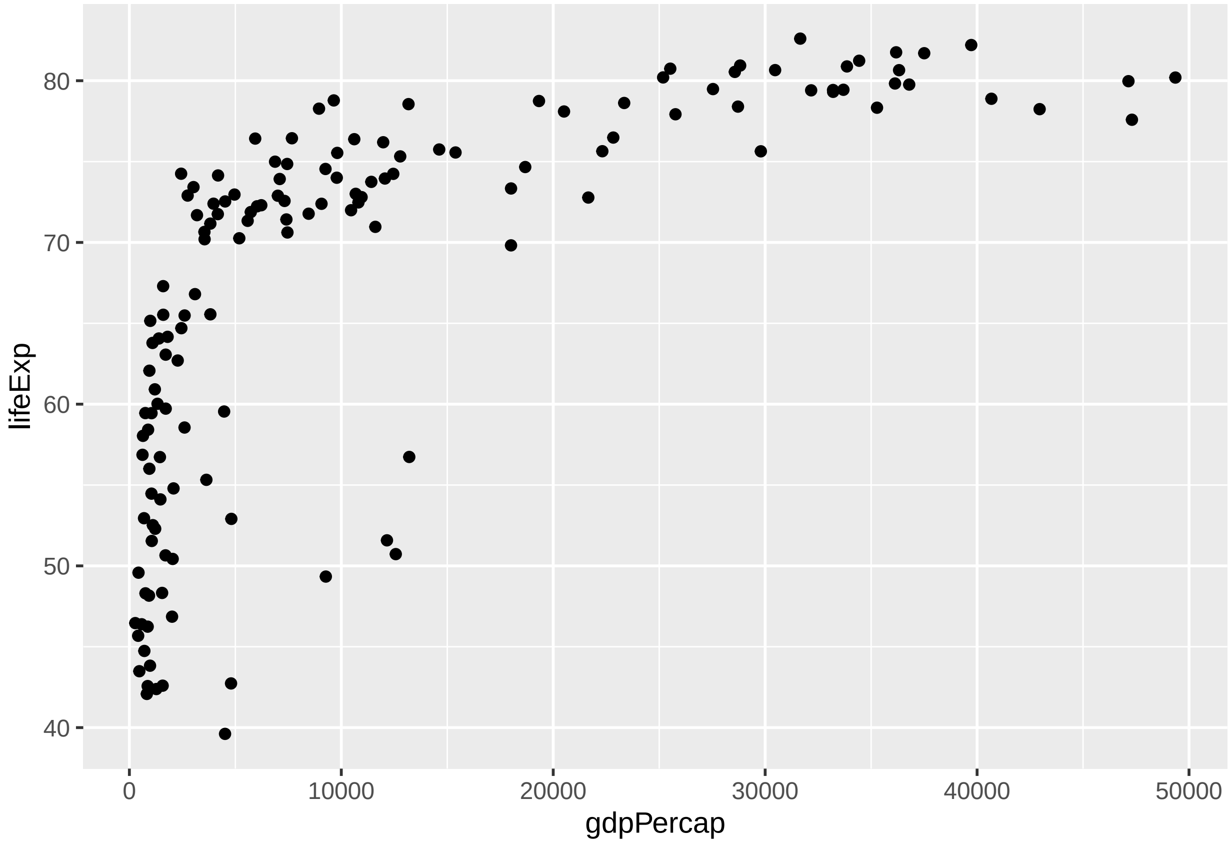


```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# OPTIONAL: Going Further
# installation & loading
# install.packages("gapminder")
library(gapminder)
# alternatively, use URL to install

# install.packages("dplyr")  ## uncoment to install if needed
library(dplyr)
```

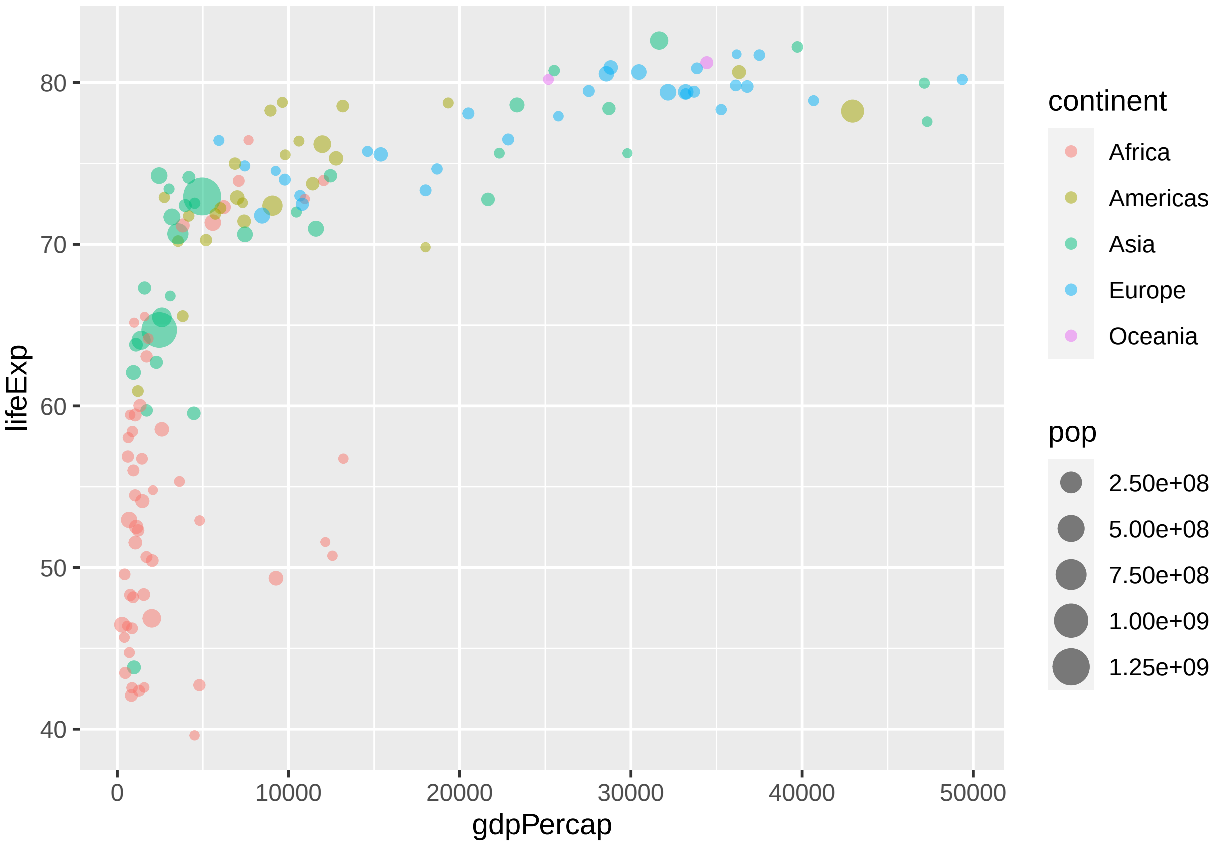
```
##
## Attaching package: 'dplyr'

## 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)
```

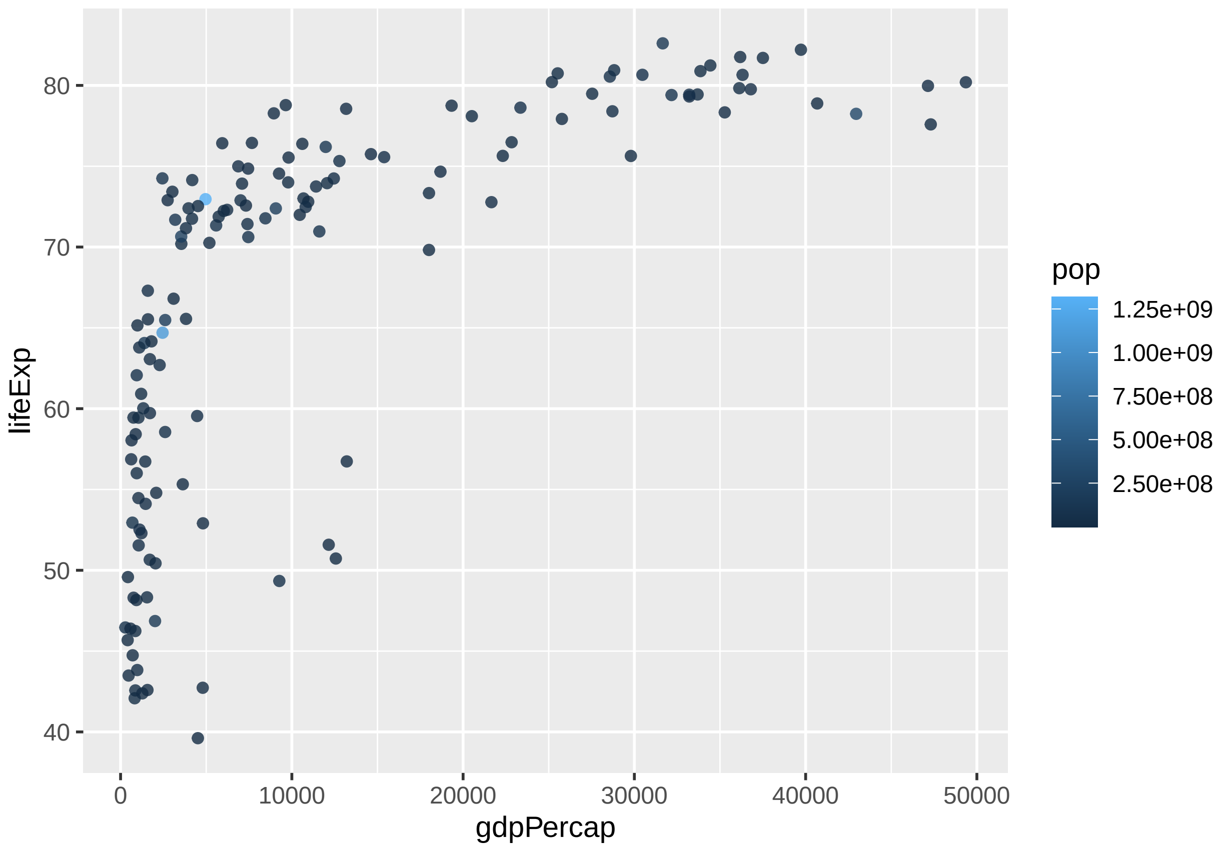
```
# Q.14 Complete the code below to produce a first basic scater plot of this
# gapminder_2007 dataset:
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point()
```



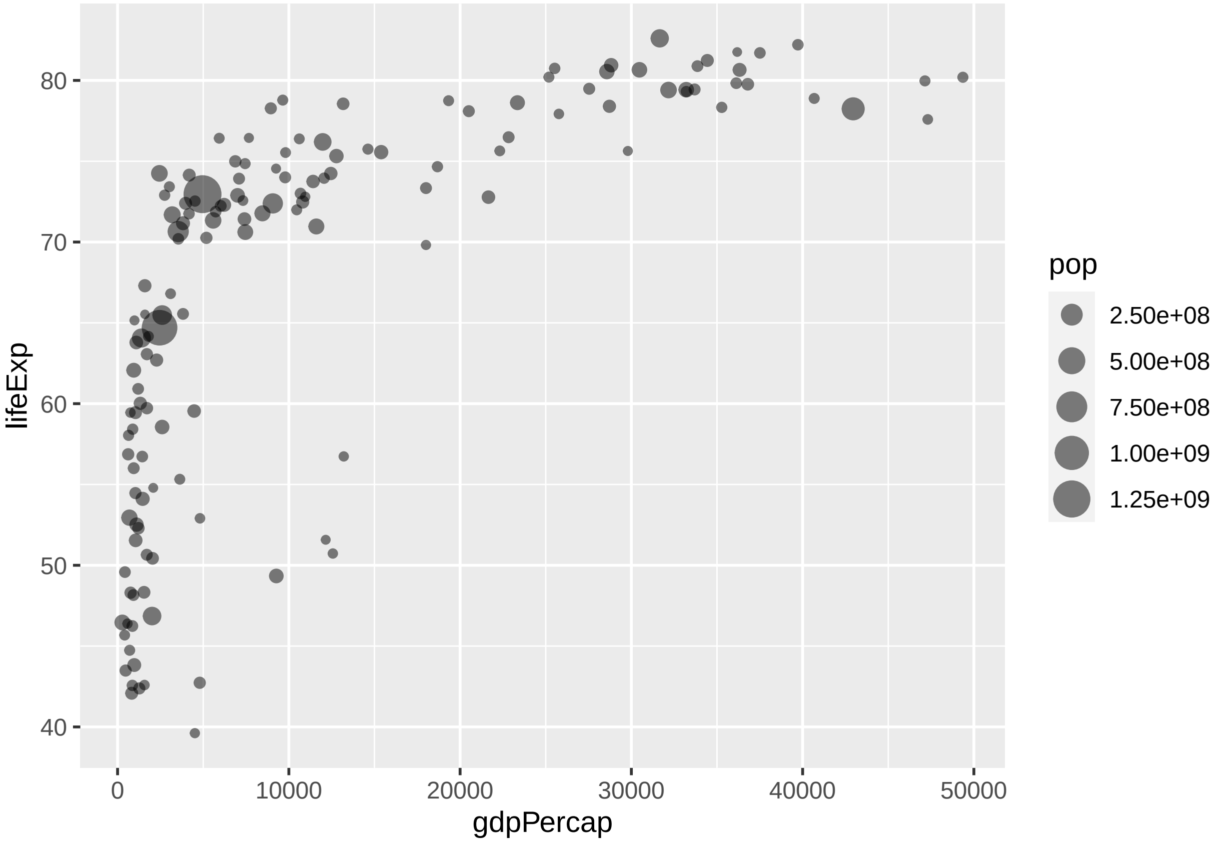
```
# adding more varables to aes()
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```



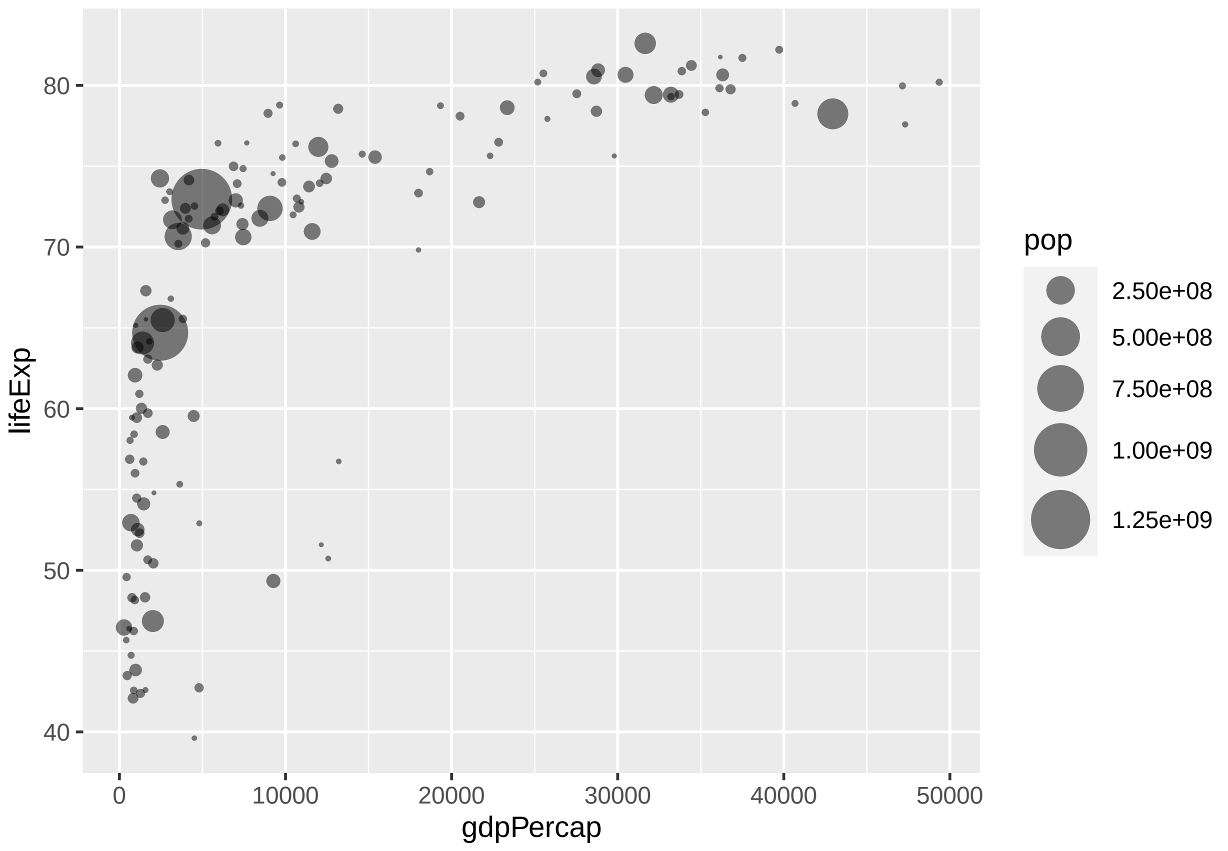
```
# color by pop
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, color = pop) +
  geom_point(alpha=0.8)
```



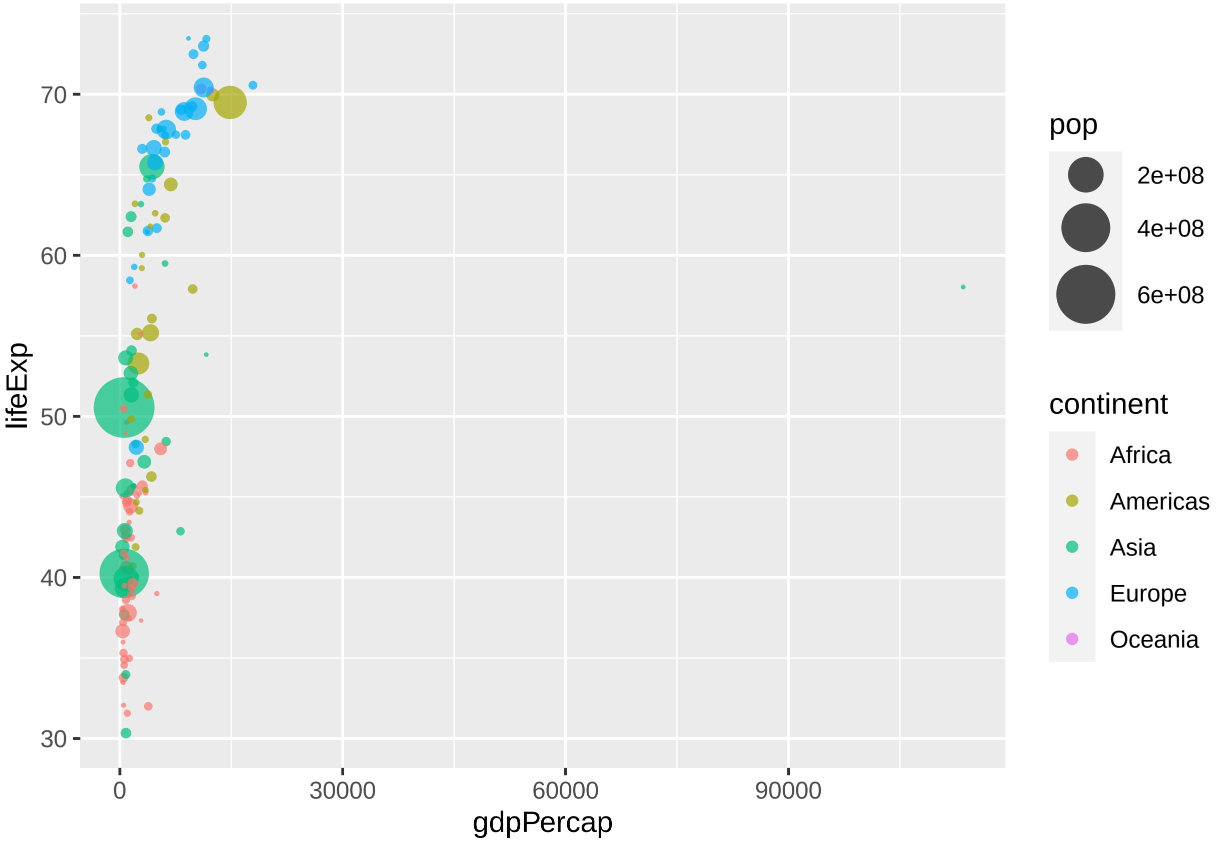
```
# adjusting point size
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, size = pop) +
  geom_point(alpha=0.5)
```



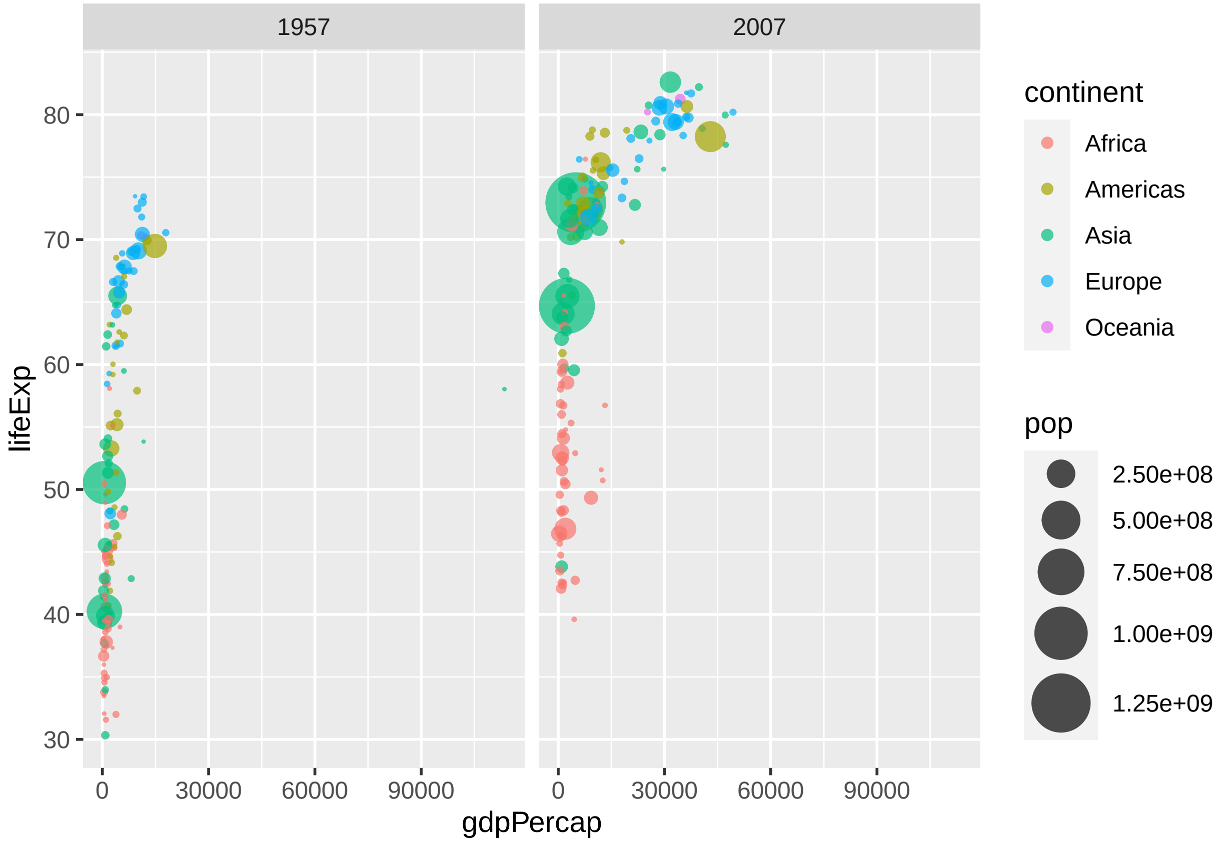
```
# changing scale
ggplot(gapminder_2007) +
  geom_point(aes(x = gdpPercap, y = lifeExp,
                 size = pop), alpha=0.5) +
  scale_size_area(max_size = 10)
```



```
# Q.15 Can you addapt the code you have leaqrned thus far to reproduce our
# gapminder scatter plot for the year 1957? What do you notice abouyt this
# plot is it easy to compare with the one for 2007?
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 = 10)
```

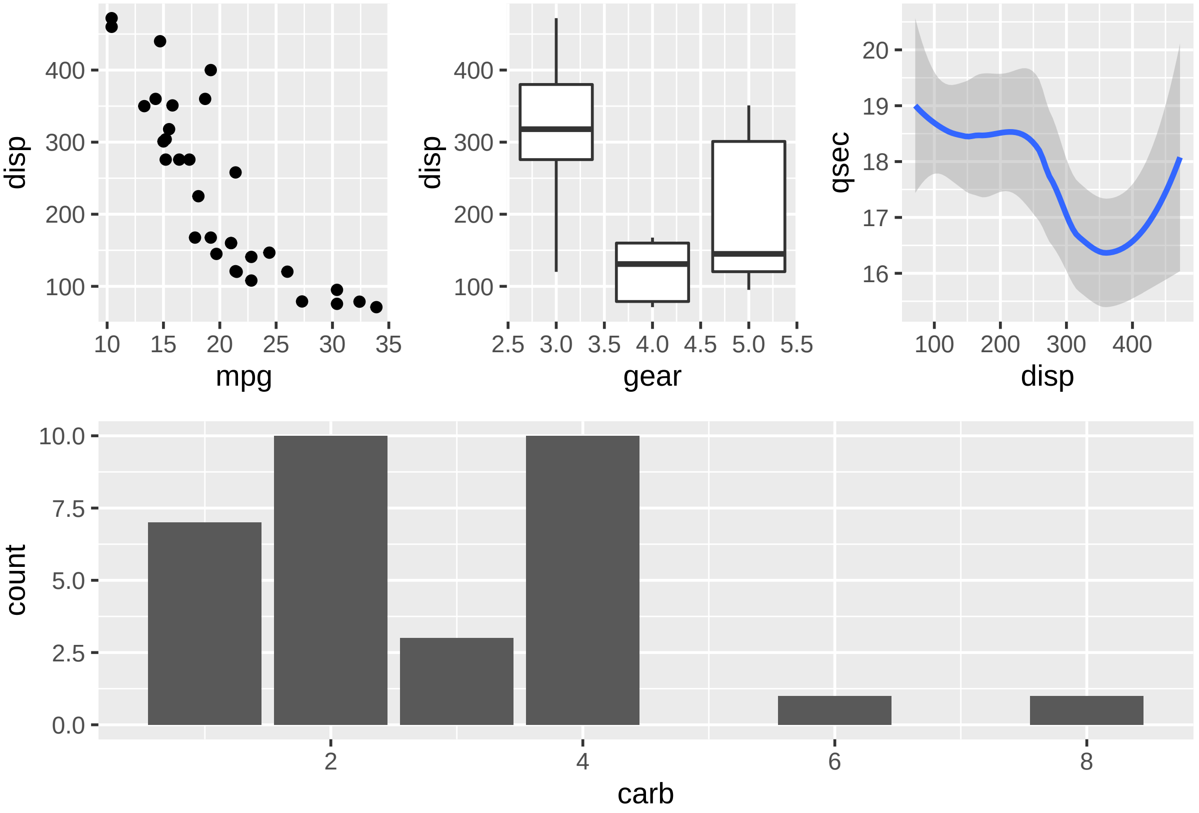


```
# Q.16 Do the same steps above but include 1957 and 2007 in your input dataset
# for ggplot(). You should now include the layer facet_wrap(~year) to produce
# the following plot:
gapminder_1957 <- gapminder %>% filter(year==1957 | year==2007)
ggplot(gapminder_1957) +
  geom_point(aes(x = gdpPercap, y = lifeExp, color=continent,
                 size = pop), alpha=0.7) +
  scale_size_area(max_size = 10) +
  facet_wrap(~year)
```



```r
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# Combining Plots (example)
# install.packages("patchwork")
library(patchwork)

# 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’
```



```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
# Session info
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] patchwork_1.1.1 dplyr_1.0.7     gapminder_0.3.0 ggplot2_3.3.5
##
## loaded via a namespace (and not attached):
##  [1] pillar_1.7.0     compiler_4.1.2   highr_0.9        tools_4.1.2
##  [5] digest_0.6.29    lattice_0.20-45  nlme_3.1-153     evaluate_0.14
```

```
##  [9] lifecycle_1.0.1  tibble_3.1.6     gtable_0.3.0     mgcv_1.8-38
## [13] pkgconfig_2.0.3  rlang_1.0.1      Matrix_1.3-4     cli_3.1.1
## [17] yaml_2.2.2       xfun_0.29        fastmap_1.1.0    withr_2.4.3
## [21] stringr_1.4.0    knitr_1.37       generics_0.1.2   vctrs_0.3.8
## [25] grid_4.1.2       tidyselect_1.1.1 glue_1.6.1       R6_2.5.1
## [29] fansi_1.0.2      rmarkdown_2.11   purrr_0.3.4      farver_2.1.0
## [33] magrittr_2.0.2   splines_4.1.2    scales_1.1.1     ellipsis_0.3.2
## [37] htmltools_0.5.2  colorspace_2.0-2 labeling_0.4.2   utf8_1.2.2
## [41] stringi_1.7.6    munsell_0.5.0    crayon_1.4.2
```