class5:Data Viz with ggplot

Bryn

#Graphs and plots in R

R has tons of different graphics systems. These include **"base R"** (e.g. the`plot()` function) and add on packages like **ggplot2**.

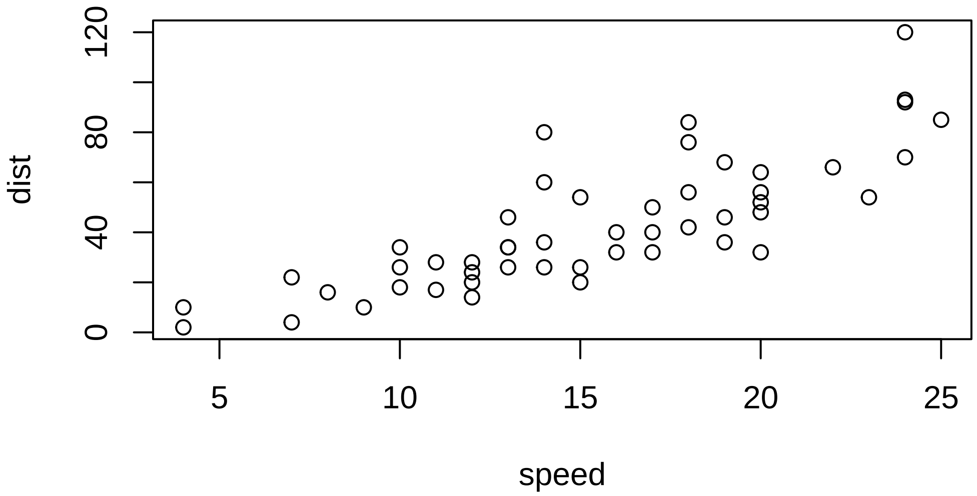
Let's start with plotting a simple dataset in "base R" and then ggplot2 to see how they differ.

```
head(cars)
```

```
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

To plot this in base R just use `plot()`

```
plot(cars)
```

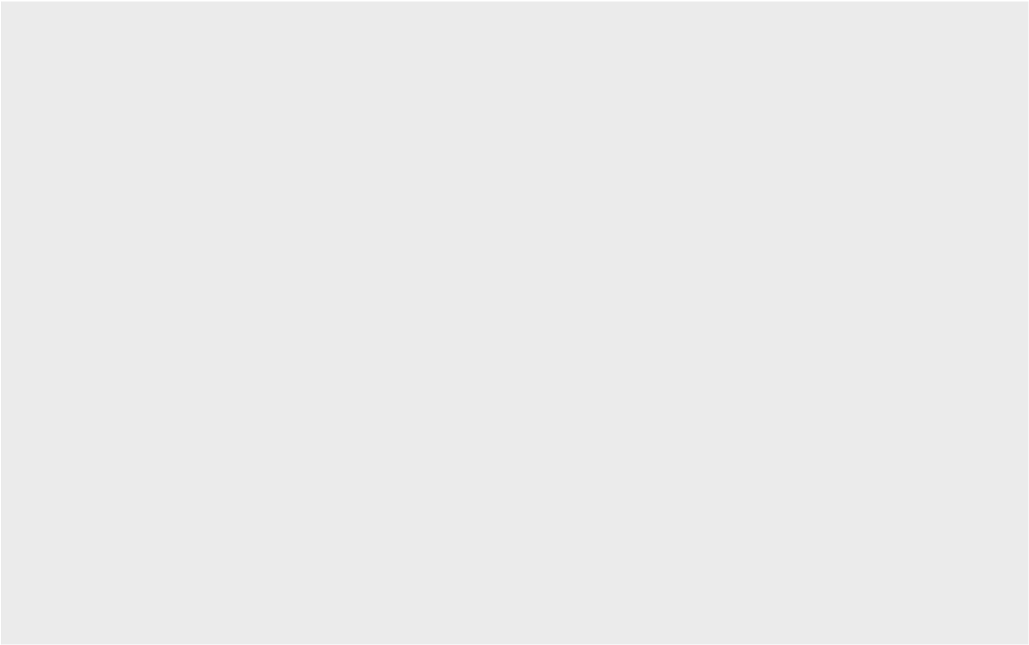


First to use ggplot2 I need to install the package. For this i use the `install.packages()` command.

> I dont want to run`install.packages()` in my quartodoc as this would re-install the package every time I render the document.

The main function in the ggplot2 package is `ggplot()`. Before I can use ggplot2 function, I need to load the package with a `library()` command.

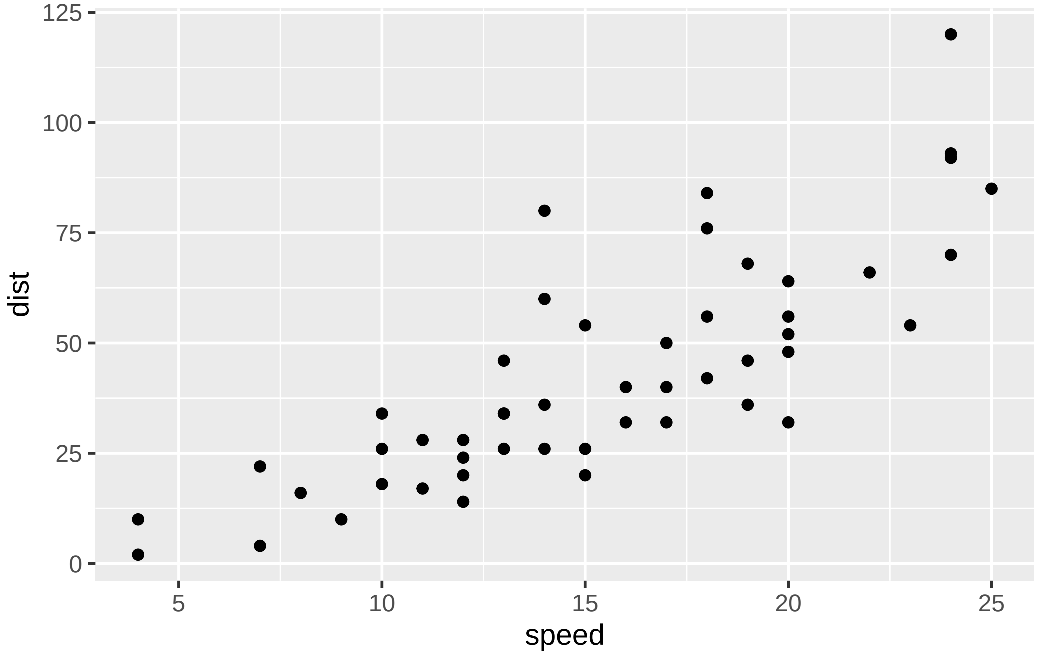
```
library(ggplot2)
ggplot(cars)
```



There are at least 3 things that every ggplot needs:

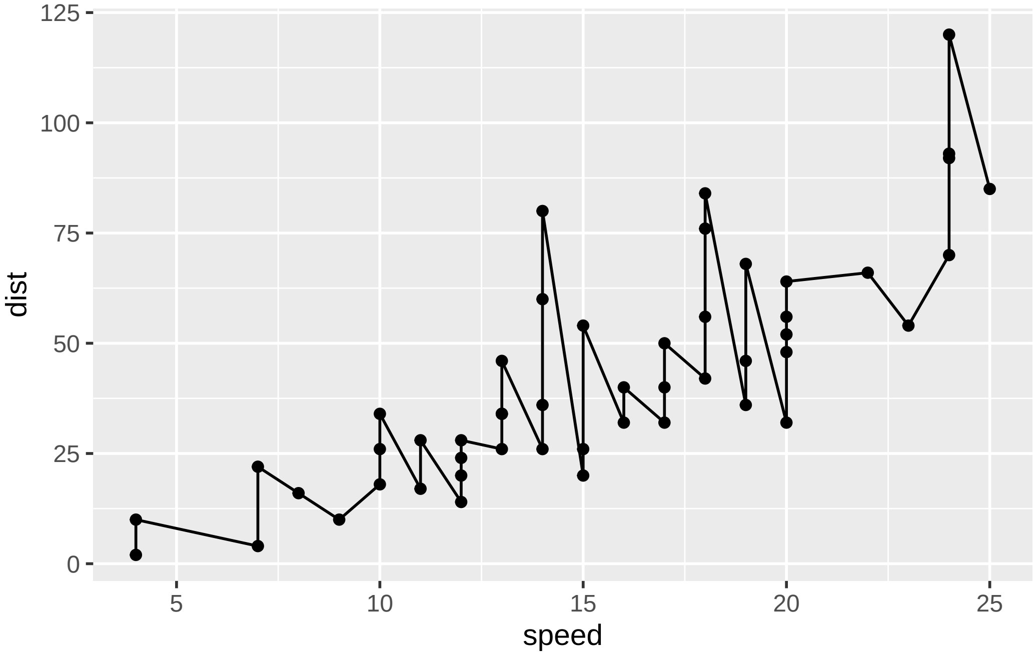
-the **data** (actual data set I want to plot) -the **aes**thetics (how the data maps to my plot) -the **geoms** or geometrics (the style of the plot)

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



I can add more layers to build more complicated plots:

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

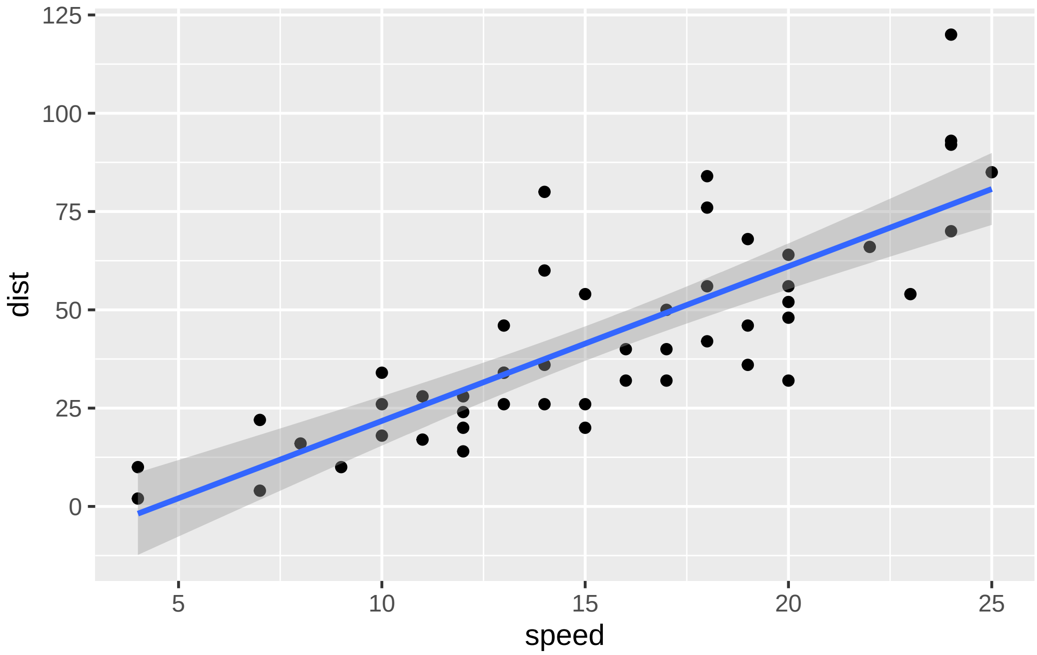


`geom_smooth()` using method

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

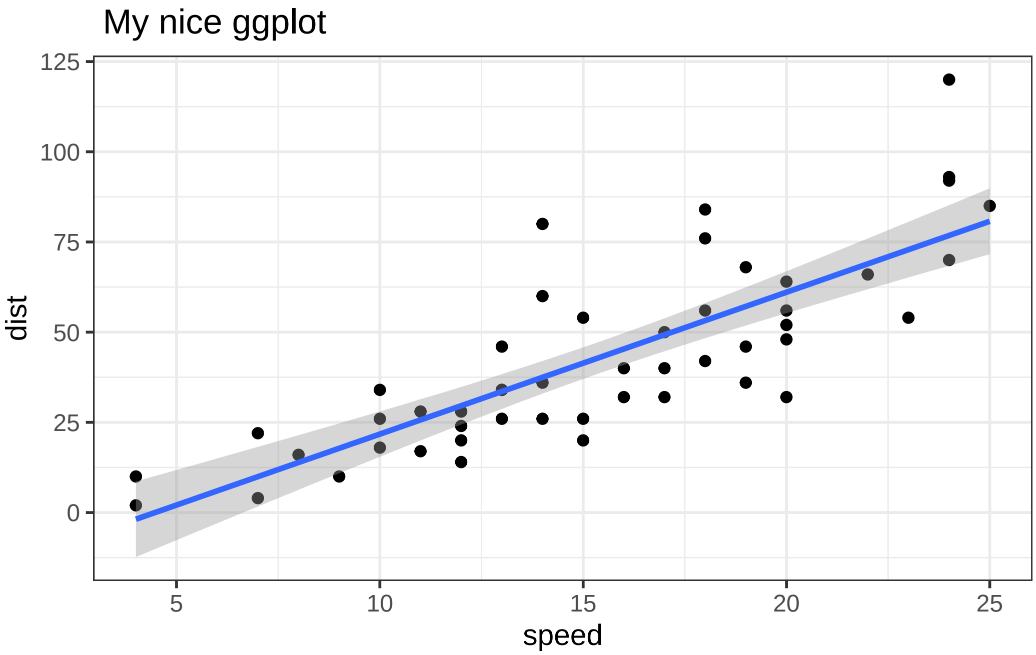
```
p
```

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



```r
p+labs(title = " My nice ggplot", X= "speed (mph)", Y= "stopping distance")+
  theme_bw()
```

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



RNA seq plot with more aes() values

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

```
nrow(genes)
```

```
[1] 5196
```

> Q. 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?

```
ncol(genes)
```

```
[1] 4
```

> Q. 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?

```
sum(genes$State=="up")
```

```
[1] 127
```

you could also do it this way:

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

```

      down unchanging         up 
        72       4997        127 
```

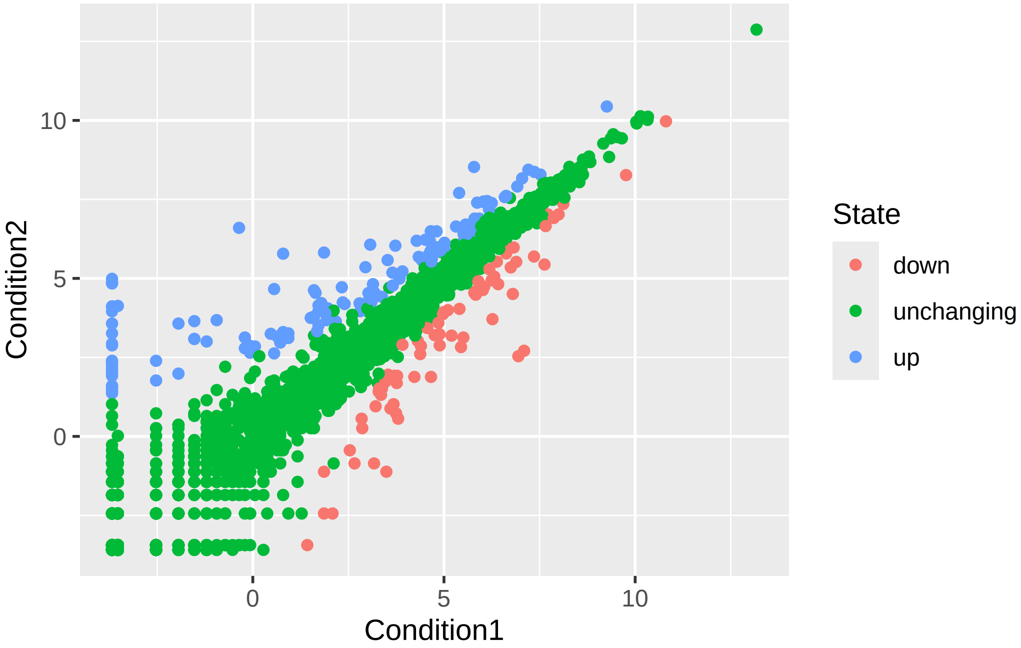
> Q. Using your values above and 2 significant figures. What fraction of total genes is up-regulated in this dataset?

```
table(genes$State)/nrow(genes) *100
```

```

      down unchanging         up 
  1.385681  96.170131   2.444188 
```

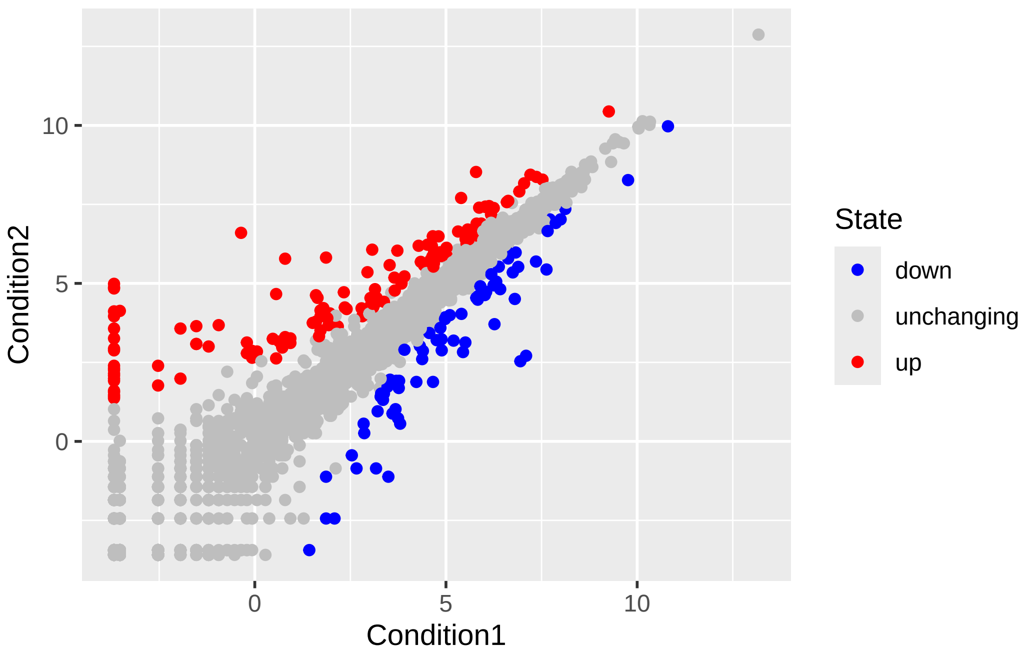
make the scatter plot with genes data:

```
b <- ggplot(genes)+
  aes(x= Condition1, y= Condition2, col=State)+
  geom_point()
b
```



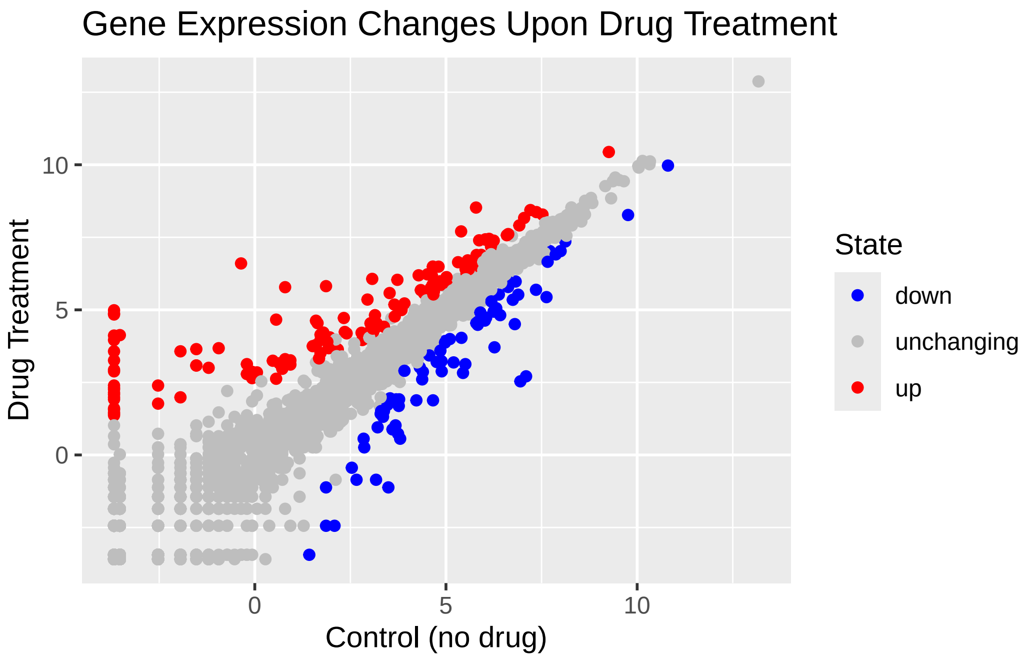
To change the colors that it defaults to use this code:

```
Newb <- b + scale_color_manual( values=c("blue", "gray","red"))

Newb
```



To add a Title and labels use this code:

```
Newb+ labs(title = "Gene Expression Changes Upon Drug Treatment",x= "Control (no drug)", y= "
```



Gapminder dataset plots

We can get exposure to setting more `aes()` parameters with datasets that include more columns of useful data. For example the **gapminder** dataset on GDP and life expendency for different countries over time.

```
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.tsv

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

Have a wee peak at the first 6 rows:

How many entries (i.e rows)

```
nrow(gapminder)
```

```
[1] 1704
```

```
head(gapminder)
```

```
      country continent year lifeExp      pop gdpPercap
1 Afghanistan      Asia 1952  28.801  8425333  779.4453
2 Afghanistan      Asia 1957  30.332  9240934  820.8530
3 Afghanistan      Asia 1962  31.997 10267083  853.1007
4 Afghanistan      Asia 1967  34.020 11537966  836.1971
5 Afghanistan      Asia 1972  36.088 13079460  739.9811
6 Afghanistan      Asia 1977  38.438 14880372  786.1134
```

How many unique countries are there in this dataset?

```
length(unique(gapminder$country))
```

```
[1] 142
```

What years does the data set cover?

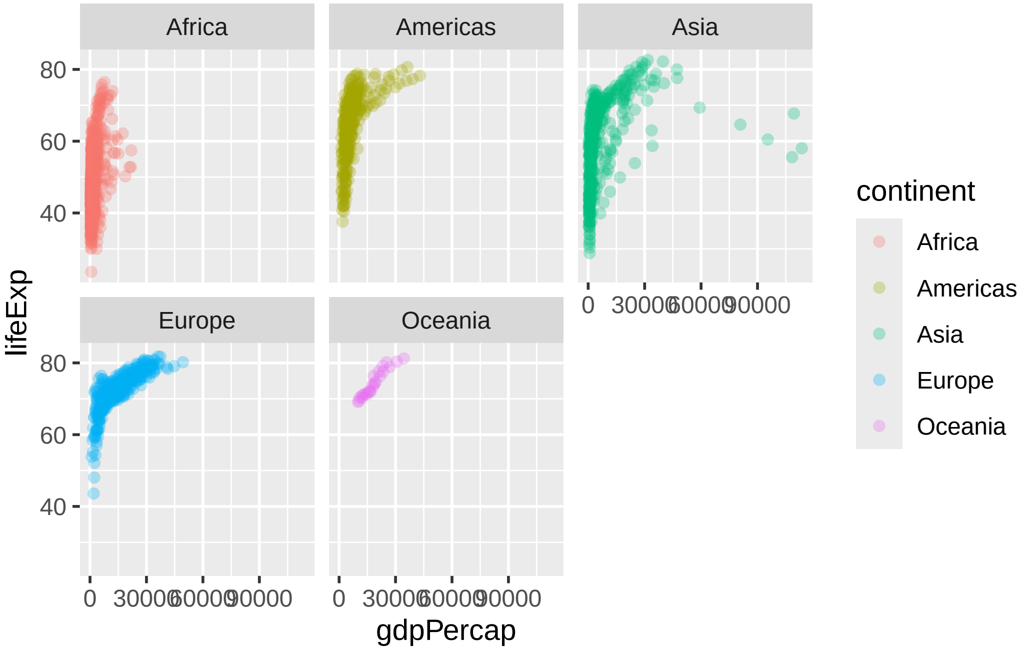
```
unique(gapminder$year)
```

```
 [1] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 2002 2007
```

Key functions that will be useful in our R journey include:

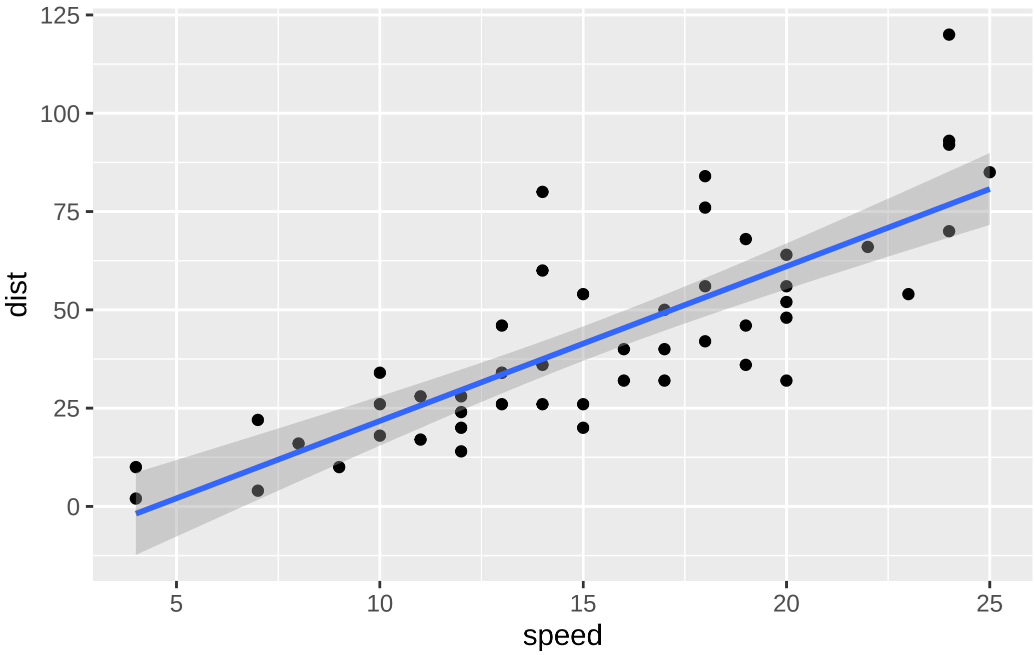
-nrow() -ncol() -length() -unique() -table()

```
c <- ggplot(gapminder)+
  aes(x=gdpPercap, y=lifeExp, col=continent)+
  geom_point(alpha=0.3)+ facet_wrap(~continent)

c
```



```
p
```

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

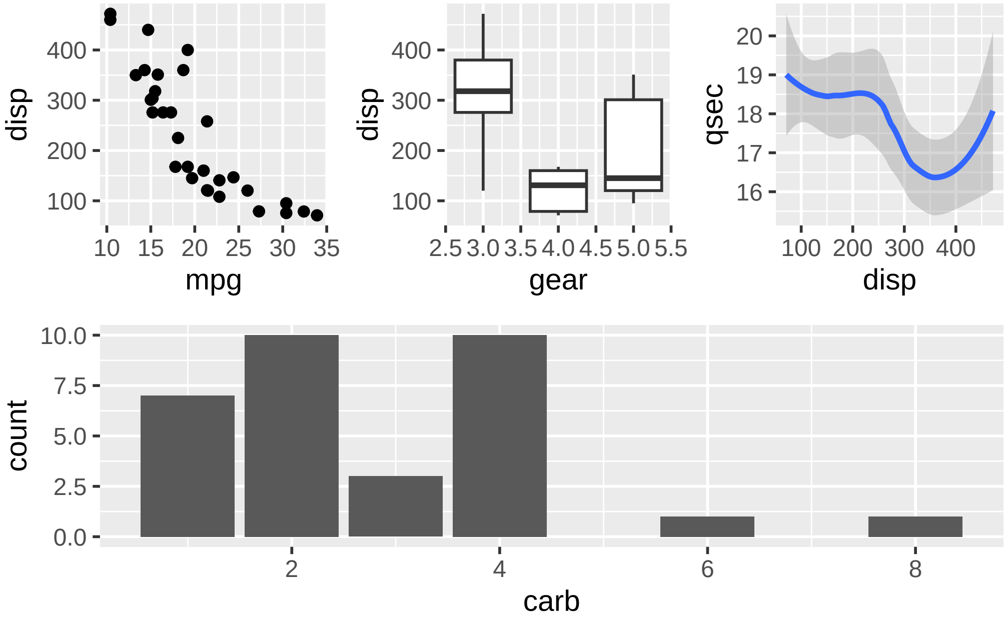


to combine plots into a figure, use the package `patchwork`

make sure to install and open with the `library(patchwork)`command:

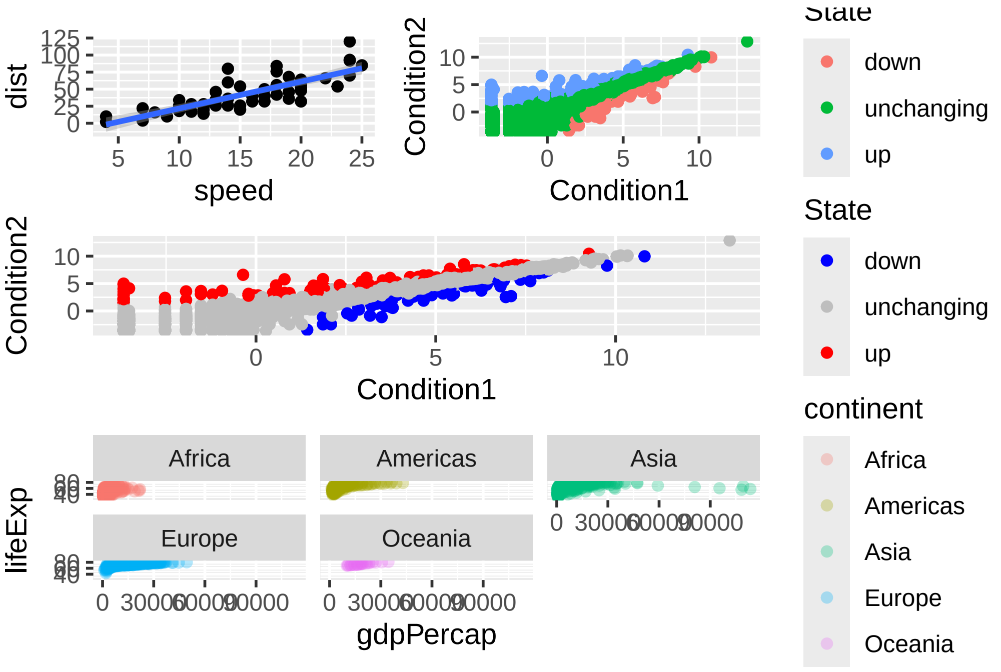
```
library(patchwork)
# Setup some example plots
p1 <- ggplot(mtcars) + geom_point(aes(mpg, disp))
p2 <- ggplot(mtcars) + geom_boxplot(aes(gear, disp, group = gear))
p3 <- ggplot(mtcars) + geom_smooth(aes(disp, qsec))
p4 <- ggplot(mtcars) + geom_bar(aes(carb))

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

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



```
(p | b) / (Newb) / (c)
```

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


dist
speed
Condition2
Condition1
State
down
unchanging
up
Condition2
Condition1
State
down
unchanging
up
Africa
Americas
Asia
Europe
Oceania
lifeExp
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania