class05: Data Viz with ggplot

Xiaoxuan Teng (PID:A69028742)

2024-01-24

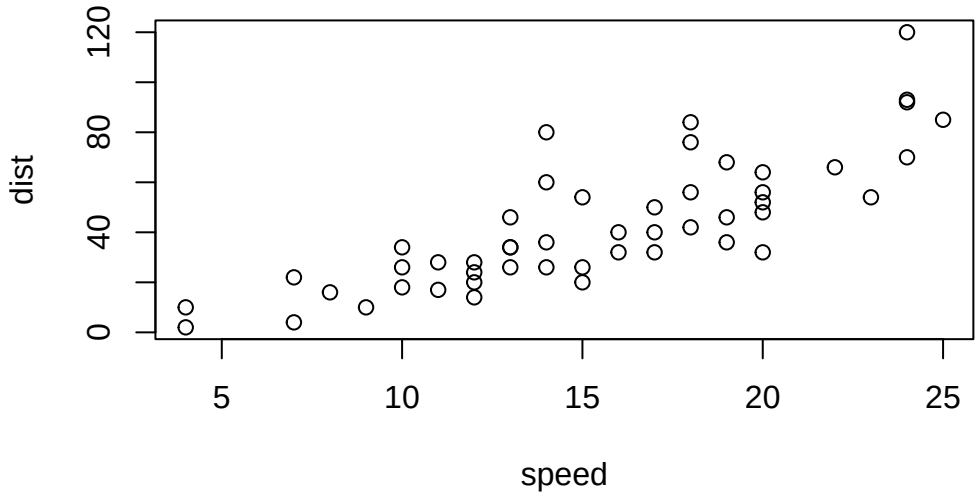
Graphics systems in R

There are many graphics systems for R. These include so-called *"base R"* and those in add-on packages like `ggplot2`.

```
#cars
str(cars)
```

```
'data.frame':   50 obs. of  2 variables:
 $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
 $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
```

```
plot(cars)
```



How can we make this with `ggplot2`

This is an add-on package and I first need to install it on my computer. This install is a one time only deal.

To install any package I use the `install.packages()` function.

To use it we need to load up the package from our library of install packages. For this I use `library(ggplot2)`

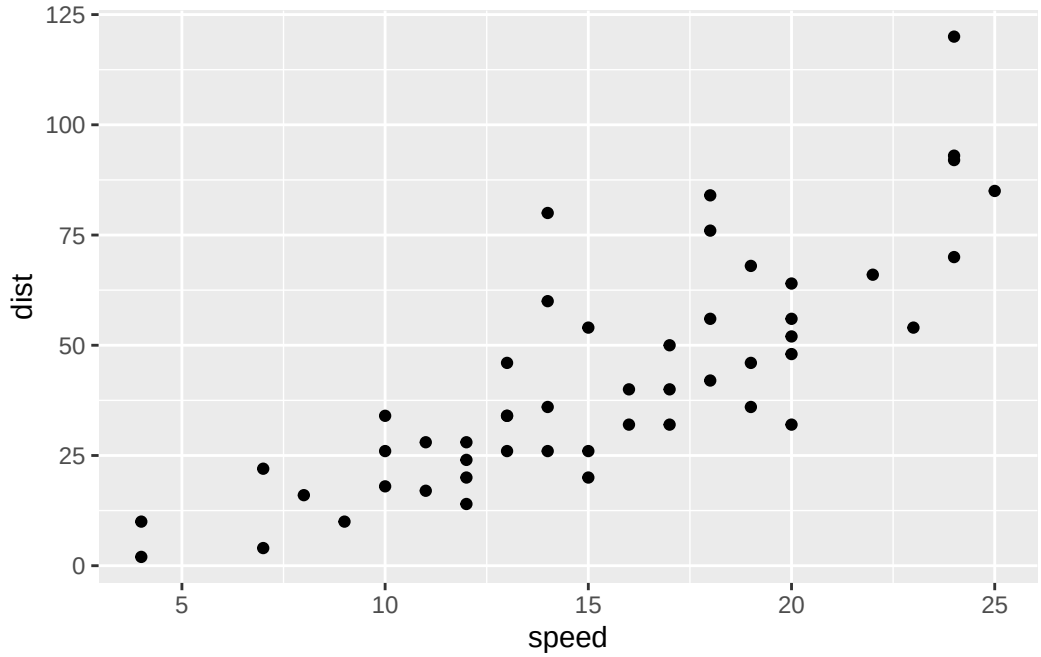
```
library(ggplot2)
ggplot(cars)
```

Using ggplot is not as straightforward as base R plot for basic plots. I have some more typing to do.

Every ggplot has at least 3 things (layers):

- **data** (data.frame)
- **aes** (how the data map to the plot)
- **geoms** (think of this as the type of plot, e.g. points, lines, etc.)

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

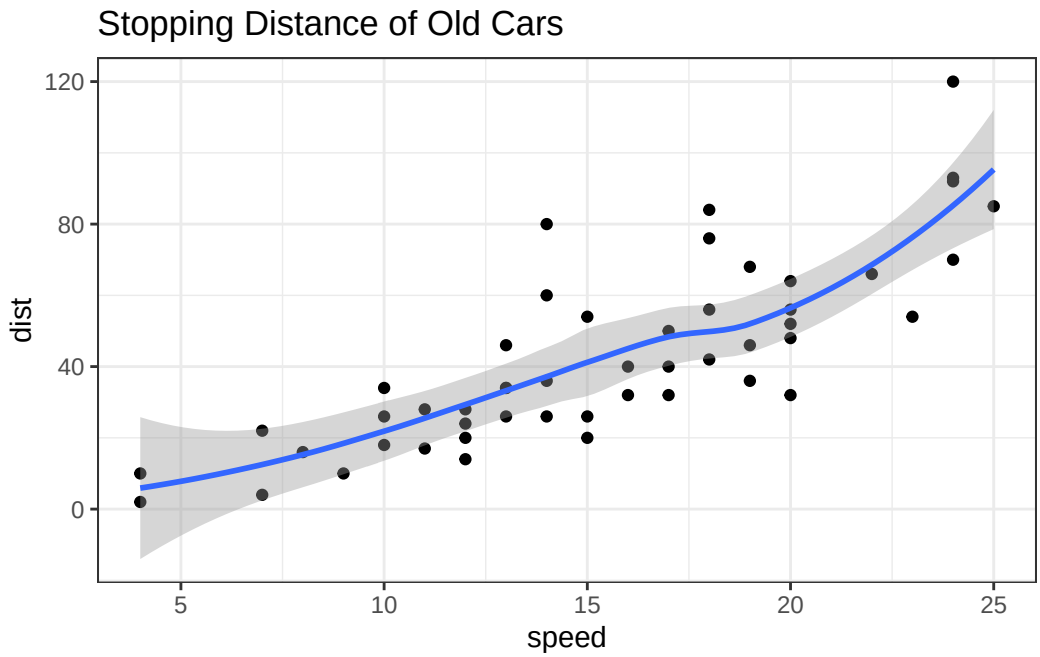



```
#ggplot(cars, aes(speed, dist, color = dist)) +
#          geom_point()
```

Here ggplot was more verbose - i.e. I had more typing to do - than base R. However, I can add more layers to make nicer and more complicated plots in an easy way with ggplot.

```
ggplot(cars) +
  aes(speed, dist) +
  geom_point() +
  geom_smooth() +
  labs(title = "Stopping Distance of Old Cars") +
  theme_bw()
```

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

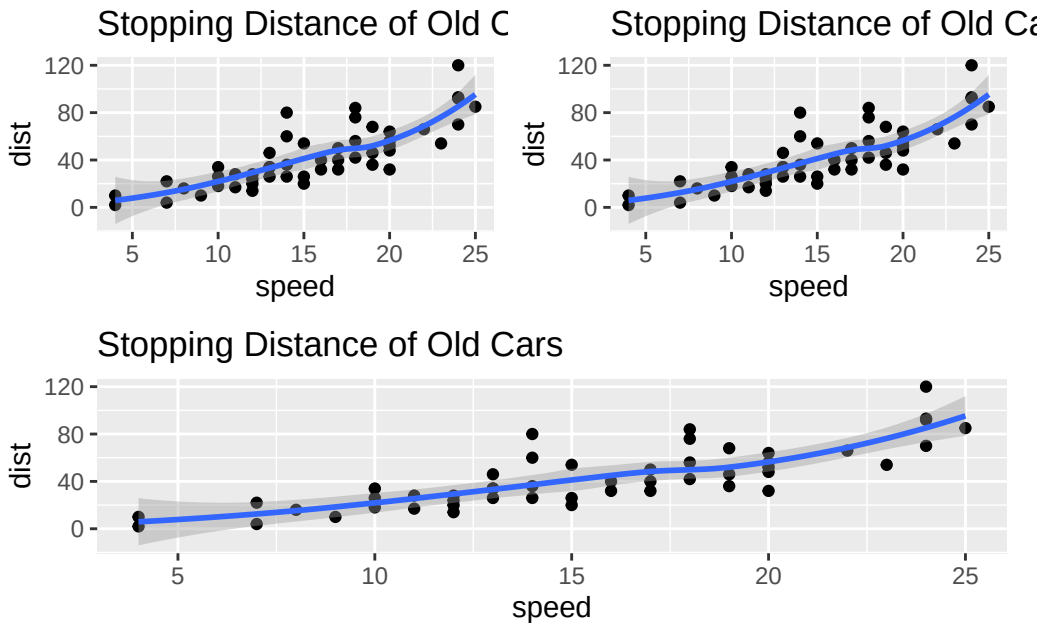



```
p <- ggplot(cars) +
  aes(speed, dist) +
  geom_point() +
  geom_smooth() +
  labs(title = "Stopping Distance of Old Cars")
```

```
library(patchwork)

(p|p) /p
```

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



Lab

```
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.txt"
genes <- read.delim(url)
head(genes, 2)
```

```
   Gene Condition1 Condition2      State
1 A4GNT  -3.680861  -3.440135 unchanging
2  AAAS   4.547958   4.386413 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?

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

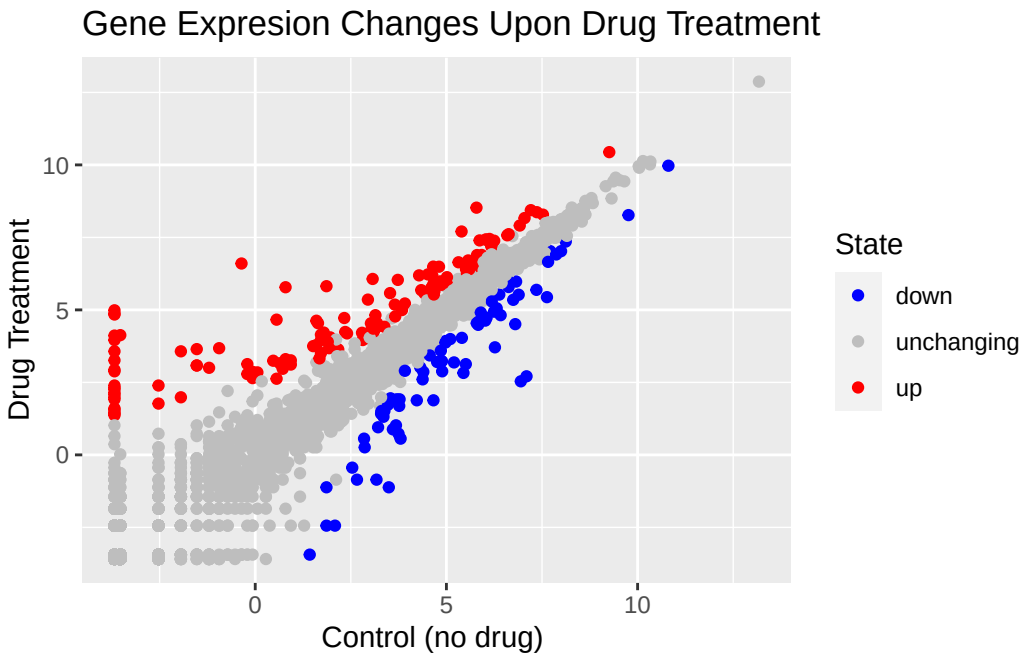
```
      down unchanging         up
        72       4997        127
```

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

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

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

```
library(ggplot2)
p <- ggplot(genes) +
    aes(x=Condition1, y=Condition2, col=State, name=Gene) +
    geom_point()
p2 <- p + scale_color_manual( values=c("blue","gray","red") ) +
  labs(title="Gene Expresion Changes Upon Drug Treatment",
         x="Control (no drug) ",
         y="Drug Treatment")
p2
```




```
library(plotly)
#ggplotly(p2)
```