Lab 01 - Hello R

Alison Yao Sept 24, 2021

Load packages and data

```
library(tidyverse)
library(datasauRus)
```

Exercises

Exercise 1

The `datasaurus_dozen` file has 1846 rows and 3 columns. There are 3 variable: `dataset` , `x` , and `y` . `dataset` indicates which dataset the data is from; `x` means the x values; `y` means the y values.

Some exploration following the instructions:

```
# follow the instructions -> check stats
datasaurus_dozen %>%
  count(dataset) %>%
  print(13)
```

```
## # A tibble:
## #   13 × 2
##    dataset
##    <chr>
##  1 away
##  2 bullseye
##  3 circle
##  4 dino
##  5 dots
##  6 h_lines
##  7 high_lines
##  8 slant_down
##  9 slant_up
## 10 star
## 11 v_lines
## 12 wide_lines
## 13 x_shape
## # … with 1
## #   more
## #   variable:
## #   n <int>
```

```
# check head
datasaurus_dozen[1:5,]
```

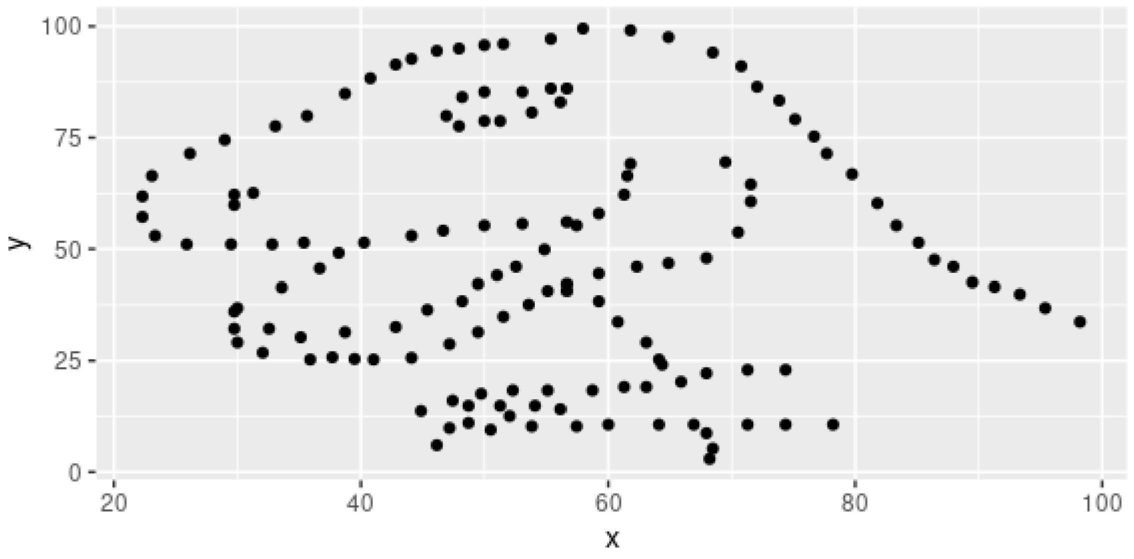
```
## # A tibble: 5 × 3
##   dataset     x     y
##   <chr>   <dbl> <dbl>
## 1 dino     55.4  97.2
## 2 dino     51.5  96.0
## 3 dino     46.2  94.5
## 4 dino     42.8  91.4
## 5 dino     40.8  88.3
```

Exercise 2

First let’s plot the data in the dino dataset:

```
dino_data <- datasaurus_dozen %>%
  filter(dataset == "dino")

ggplot(data = dino_data, mapping = aes(x = x, y = y)) +
  geom_point()
```



And next calculate the correlation between `x` and `y` in this dataset:

```
dino_data %>%
  summarize(r = cor(x, y))
```

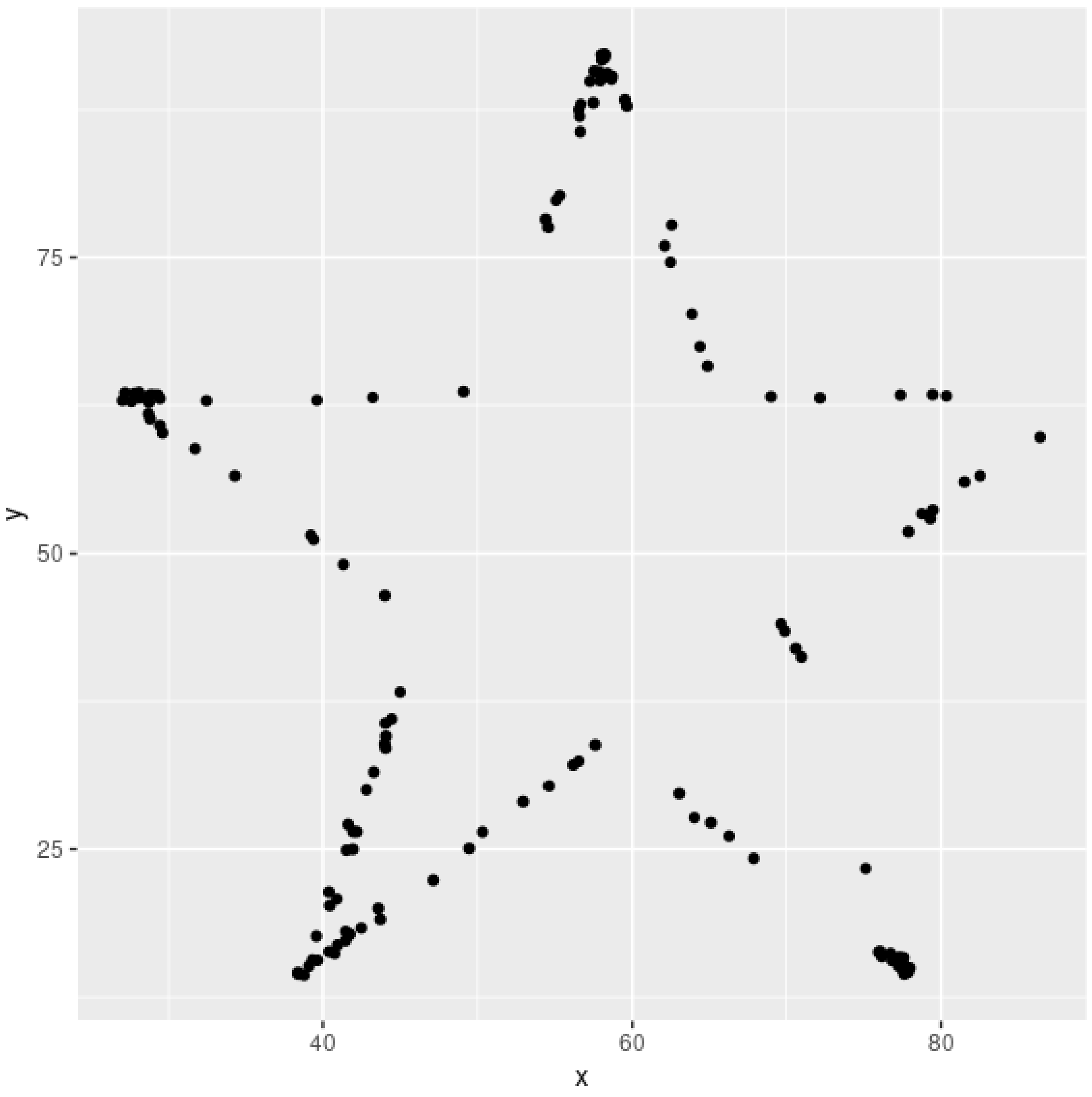
```
## # A tibble: 1 × 1
##         r
##     <dbl>
## 1 -0.0645
```

Exercise 3

Similarly, we first filter the observations belonging to the star dataset and store the filtering results into a new dataframe called `star_data`. Then, we use `ggplot` to visualize the data in points.

```
star_data <- datasaurus_dozen %>%
  filter(dataset == "star")

ggplot(data = star_data, mapping = aes(x = x, y = y)) +
  geom_point()
```



Again, we use the pipe operator `%>%` to send the dataframe `star_data` as the first argument to the `summarize` function and calculate the correlation coefficient.

```
star_data %>%
  summarize(r = cor(x, y))
```

```
## # A tibble: 1 × 1
##         r
##     <dbl>
## 1 -0.0630
```

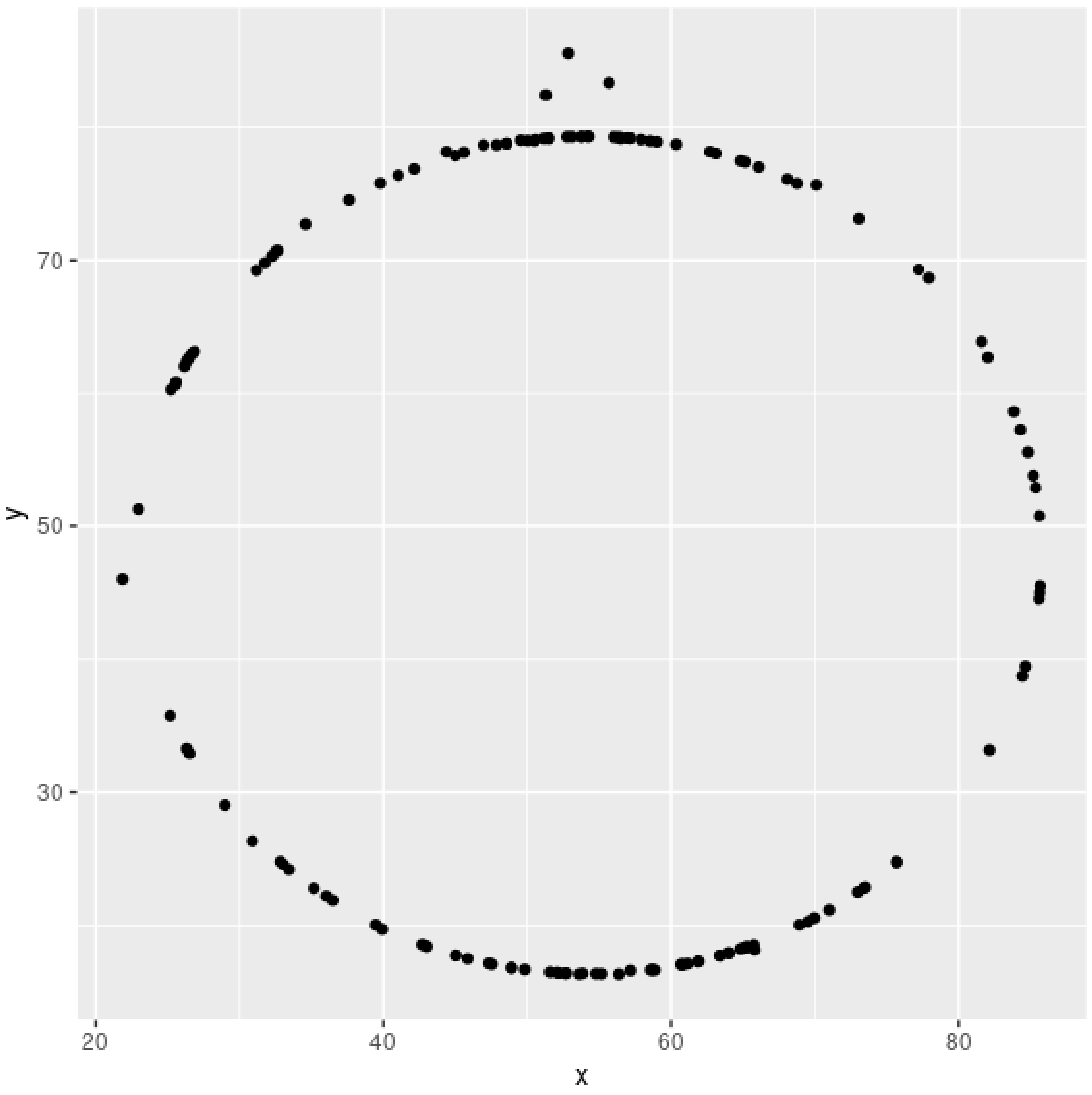
Compared to the `r` of `dino`, the `r` of `star` is slightly bigger, but almost the same.

Exercise 4

We filter the observations belonging to the `circle` dataset and store the filtering results into a new dataframe called `circle_data`. Then, we use `ggplot` to visualize the data in points.

```
circle_data <- datasaurus_dozen %>%
  filter(dataset == "circle")

ggplot(data = circle_data, mapping = aes(x = x, y = y)) +
  geom_point()
```



We can use the pipe operator `%>%` to send the dataframe `circle_data` as the first argument to the `summarize` function and calculate its correlation coefficient accordingly.

```
circle_data %>%
  summarize(r = cor(x, y))
```

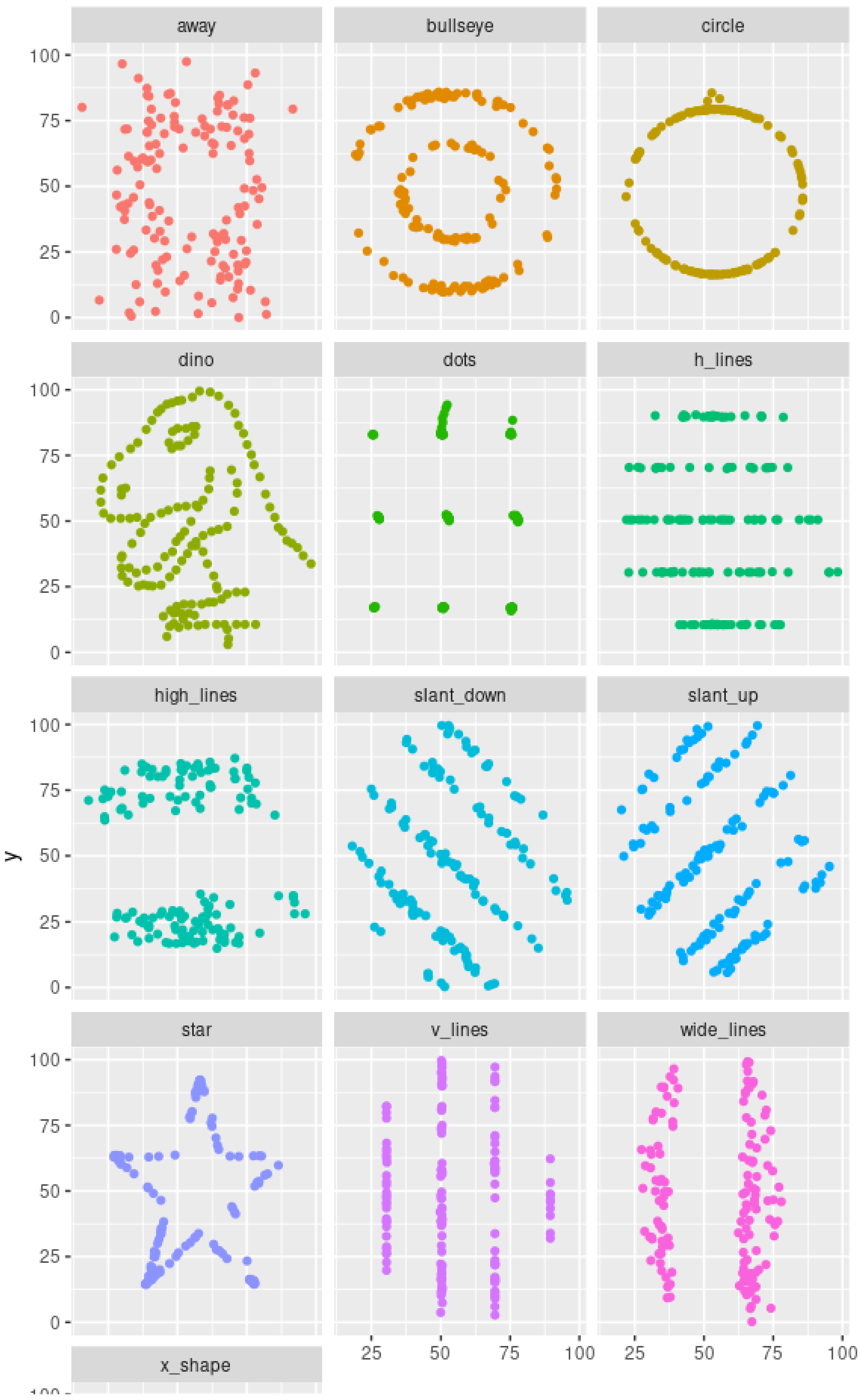
```
## # A tibble: 1 × 1
##         r
##     <dbl>
## 1 -0.0683
```

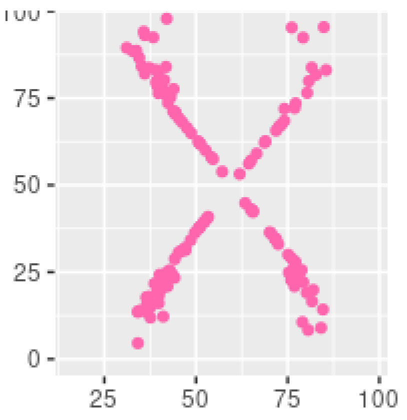
Compared to the `r` of `dino`, the `r` of `circle` is a little bit smaller by about 0.004.

Exercise 5

First, let’s use faceting to plot all datasets at once.

```r
ggplot(datasaurus_dozen, aes(x = x, y = y, color = dataset))+
  geom_point()+
  facet_wrap(~ dataset, ncol = 3) +
  theme(legend.position = "none")
```





x

Then, we use the `group_by` function to generate all the summary correlation coefficients.

```
datasaurus_dozen %>%
  group_by(dataset) %>%
  summarize(r = cor(x, y)) %>%
  print(13)
```

```
## # A tibble:
## #   13 × 2
##    dataset
##    <chr>
##  1 away
##  2 bullseye
##  3 circle
##  4 dino
##  5 dots
##  6 h_lines
##  7 high_lines
##  8 slant_down
##  9 slant_up
## 10 star
## 11 v_lines
## 12 wide_lines
## 13 x_shape
## # … with 1
## #   more
## #   variable:
## #   r <dbl>
```