PM566 Lab 1

AUTHOR
Chih-Chan Jessica Lan

Input Libray and Data

```
library(datasauRus)
library(ggplot2)
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
```

```
library(tidyr)

df <- datasaurus_dozen
```

Question 1

Based on the help file, how many rows and how many columns does the datasaurus_dozen file have? What are the variables included in the data frame? Add your responses to your lab report, with relevant code in the associated R code chunk, and free-form text outside of the code chunk.

Answer:

There are 1846 rows and 3 columns in the datasaurus_dozen file. The 3 variable names include: dataset, x, and y

```
dim(df)
```

```
[1] 1846    3
```

```
# cat("Dimension of dataset:", dim(df), "\n")
# print('Names of columns in the dataset:', names(df))
```

Summary of dataset names: There are 13 datasets

```
table(datasaurus_dozen$dataset)
```

```

      away   bullseye     circle       dino       dots    h_lines high_lines 
       142        142        142        142        142        142        142 
slant_down   slant_up       star    v_lines wide_lines    x_shape 
       142        142        142        142        142        142 
```

Summary of Continuous Variables (x and y)

```
# Variable x
summary(df$x)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  15.56   41.07   52.59   54.27   67.28   98.29 
```

```
# Variable y
summary(df$y)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
 0.01512 22.56107 47.59445 47.83510 71.81078 99.69468 
```

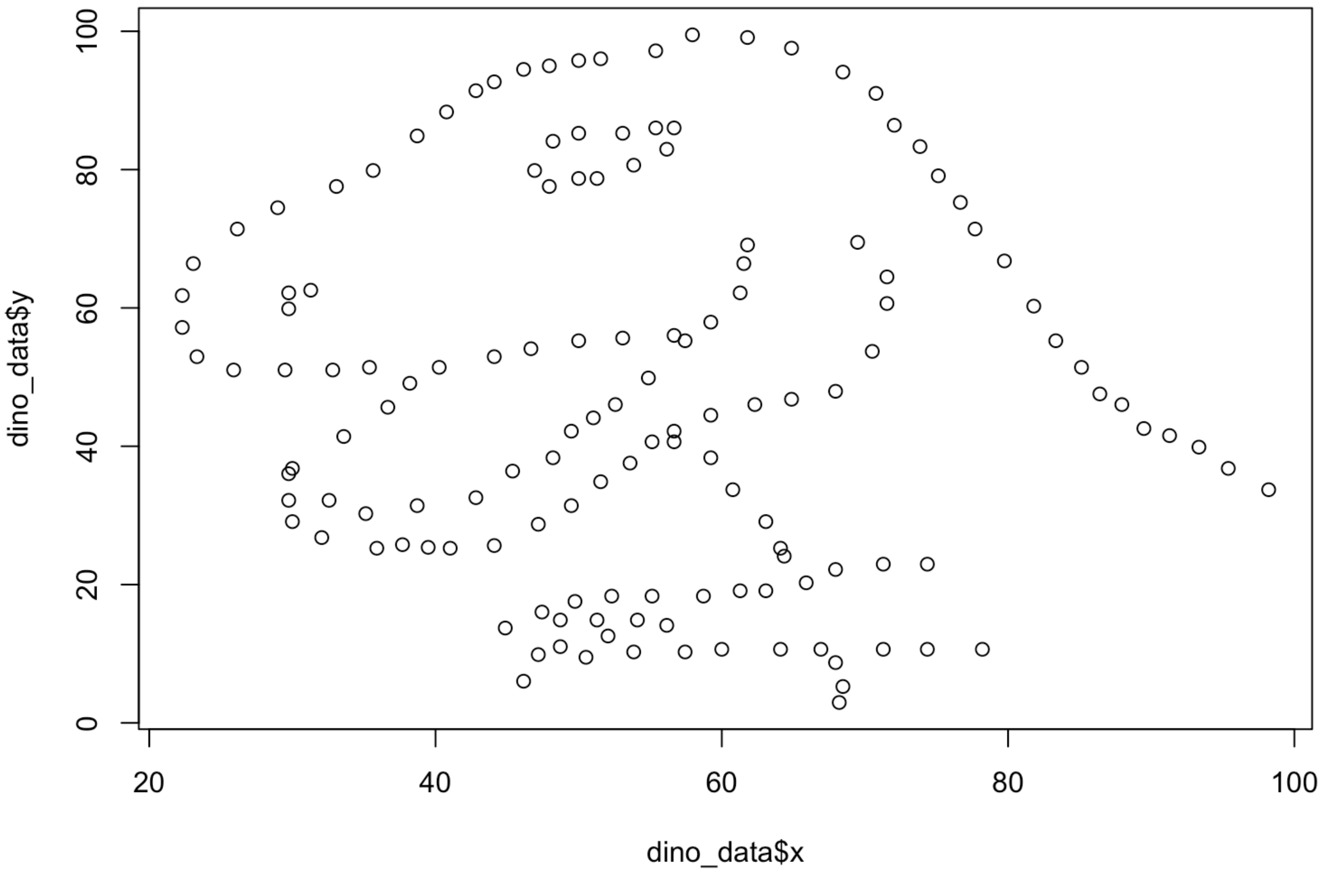
Question 2

Plot y vs. x for the dino dataset. Then, calculate the correlation coefficient between x and y for just this dataset.

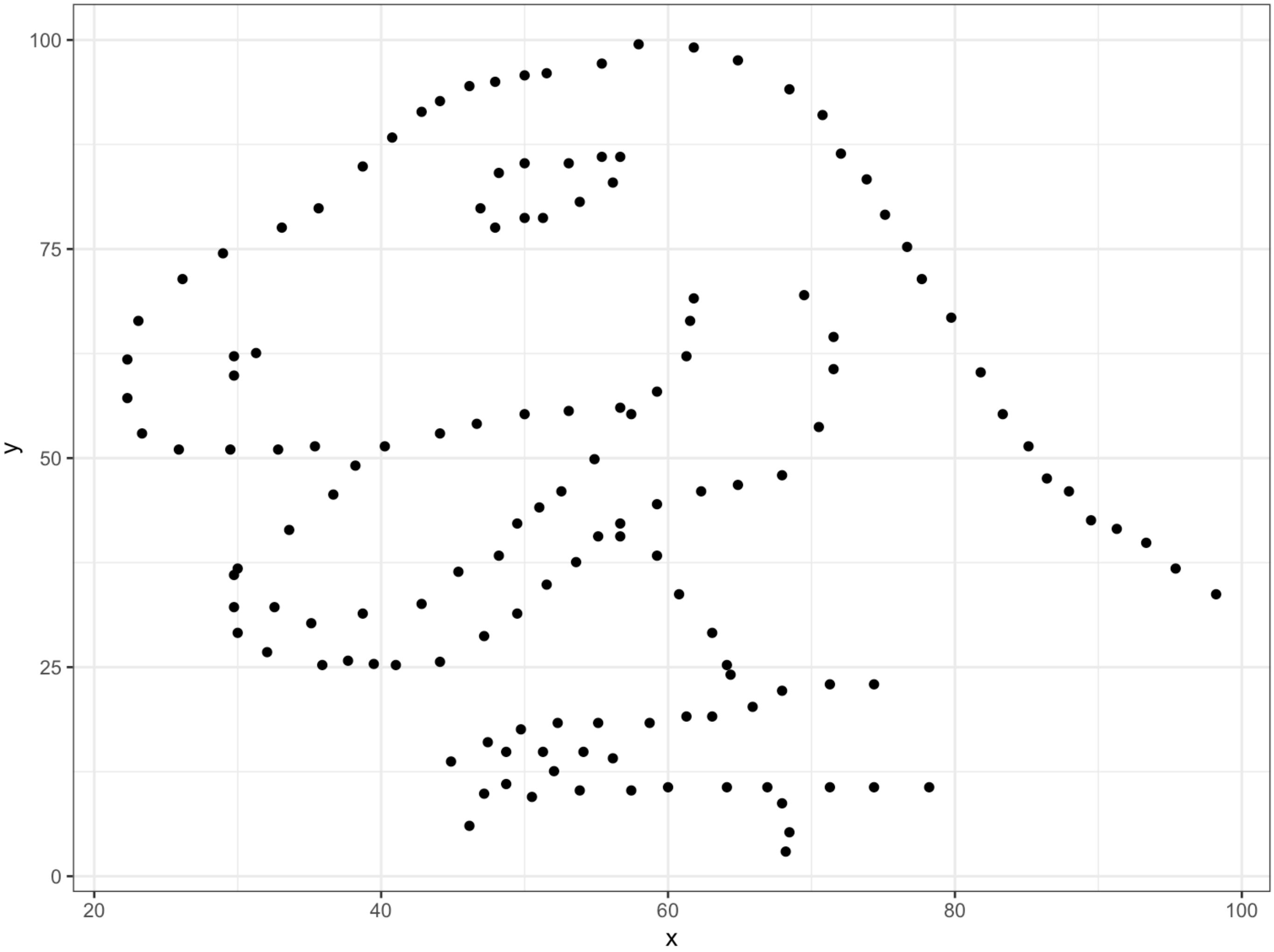
Plot

```
dino_data <- df[df$dataset == 'dino', ]
```

```
plot(dino_data$x, dino_data$y)
```



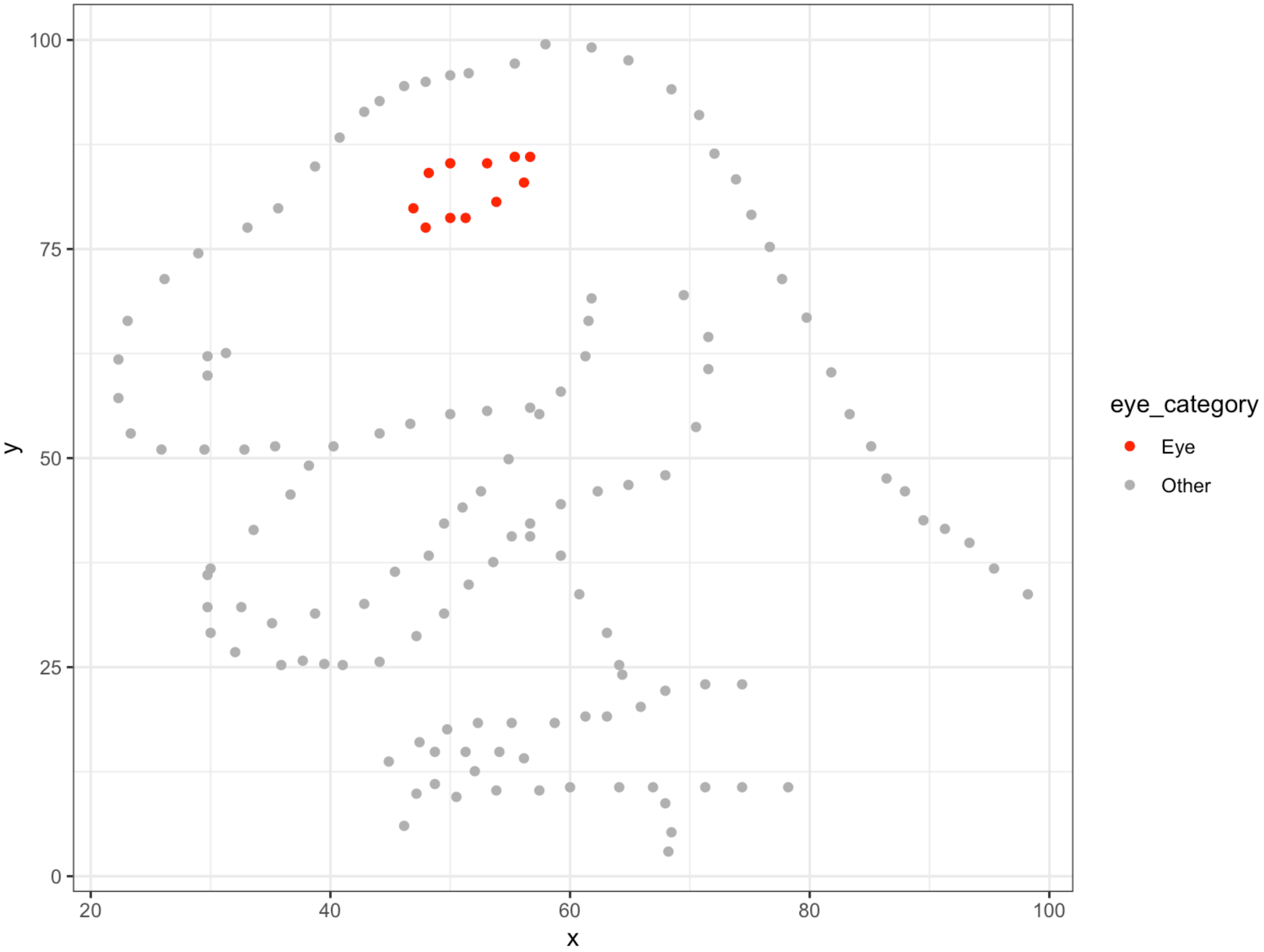
```
ggplot(data = dino_data, mapping = aes(x = x, y = y)) +
  geom_point() +
  theme_bw()
```



Reference: https://stackoverflow.com/questions/10861773/remove-grid-background-color-and-top-and-right-borders-from-ggplot2

```
dino_data_eye <- dino_data %>%
  mutate(
    eye_category = case_when(
      x >= 40 & x <= 60 & y >= 75 & y <= 88 ~ "Eye",
      TRUE ~ "Other"
    )
  )

ggplot(dino_data_eye, aes(x = x, y = y, color = eye_category)) +
  geom_point() +
  scale_color_manual(values = c("Eye" = "red", "Other" = "gray70")) +
  theme_bw()
```



Calculate correlation coefficient (r)

```
cor(dino_data$x, dino_data$y)
```

```
[1] -0.06447185
```

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

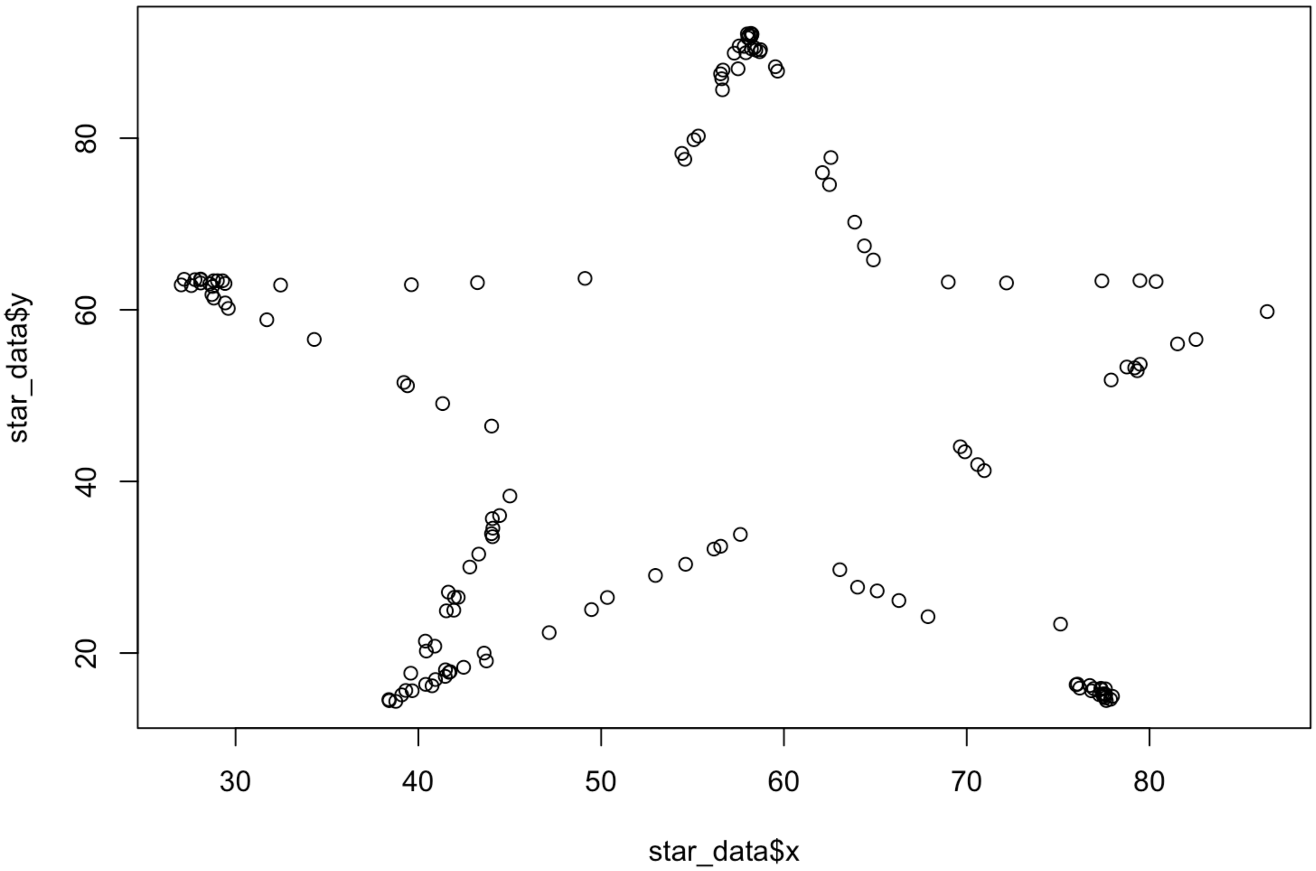
Question 3

Now try it on your own! Plot y vs. x for the star dataset, another one of the datasaurus_dozen. You can (and should) re-use code we introduced above, just replace the dataset name with the desired dataset. Then, calculate the correlation coefficient between x and y for this dataset. How does this value compare to the r of dino?

```
star_data <- df[df$dataset == "star", ]
```

###Plot

```
plot(star_data$x, star_data$y)
```



Calculate correlation coefficient (r)

```
cor(star_data$x, star_data$y)
```

```
[1] -0.0629611
```

The correlation coefficients (r) of star and dino data are almost the same around -0.06. Although the rs are similar, the visualization plots actually show a significant difference. This is showing that visualizing data is also critical when we are diving into given data.

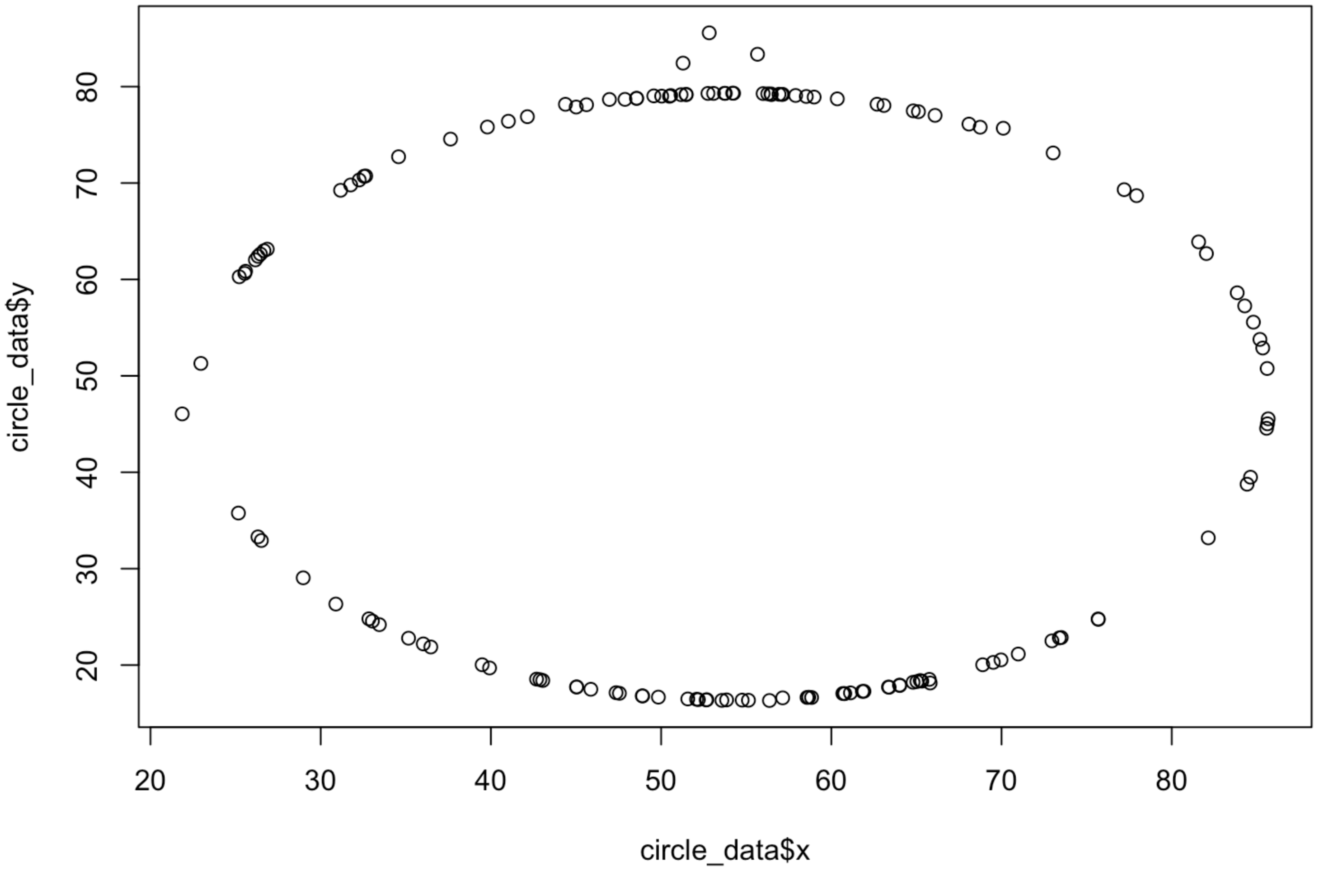
Question 4

Plot y vs. x for the circle dataset. You can (and should) reuse code we introduced above, just replace the dataset name with the desired dataset. Then, calculate the correlation coefficient between x and y for this dataset. How does this value compare to the r of dino?

```
circle_data <- df[df$dataset == "circle", ]
```

Plot

```
plot(circle_data$x, circle_data$y)
```



Calculate correlation coefficient (r)

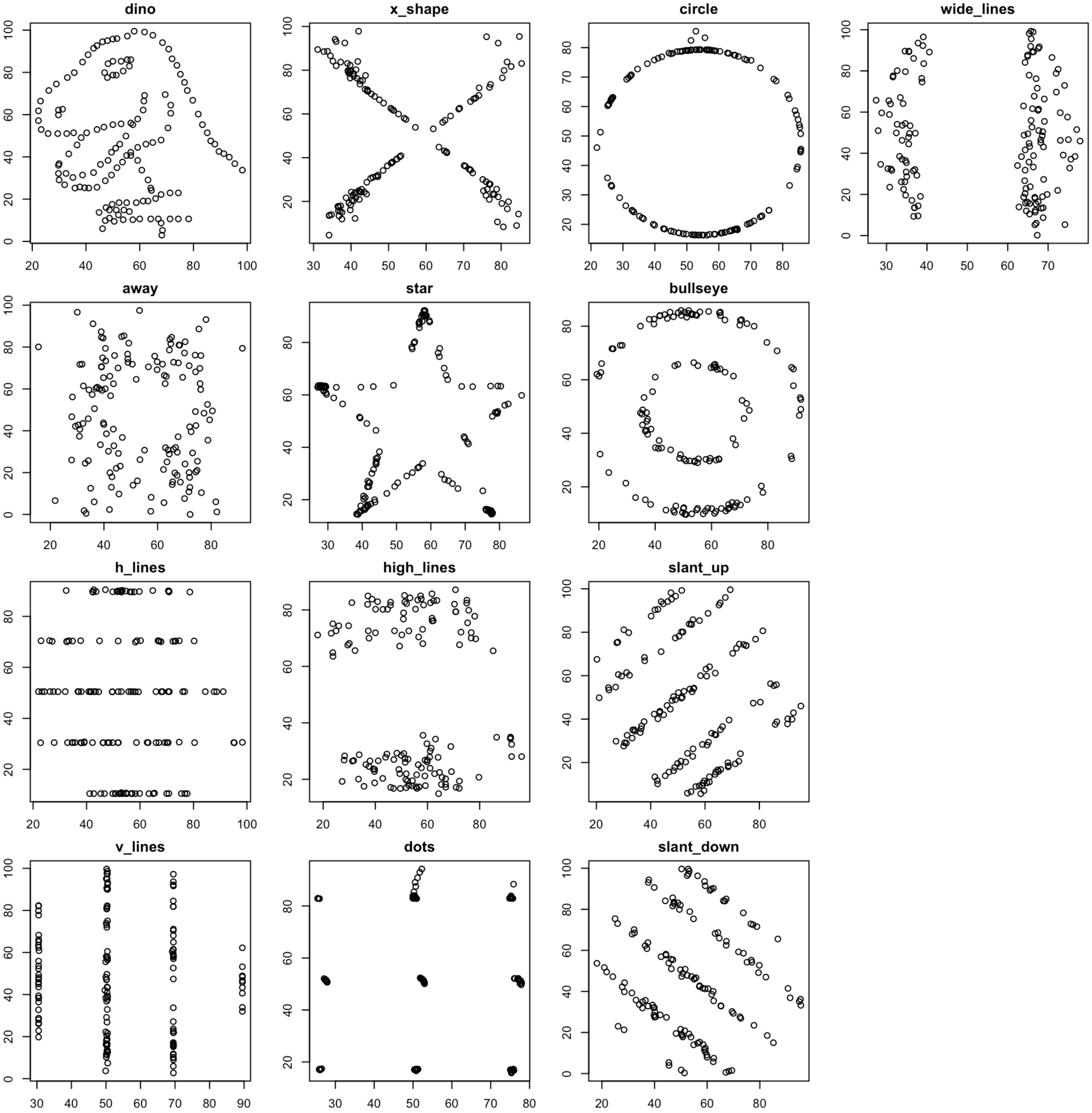
```
cor(circle_data$x, circle_data$y)
```

```
[1] -0.06834336
```

The correlation coefficients (r) of circle and dino data are almost the same around -0.06. Although the rs are similar, the visualization plots actually show a significant difference. This is showing that visualizing data is also critical when we are diving into given data.

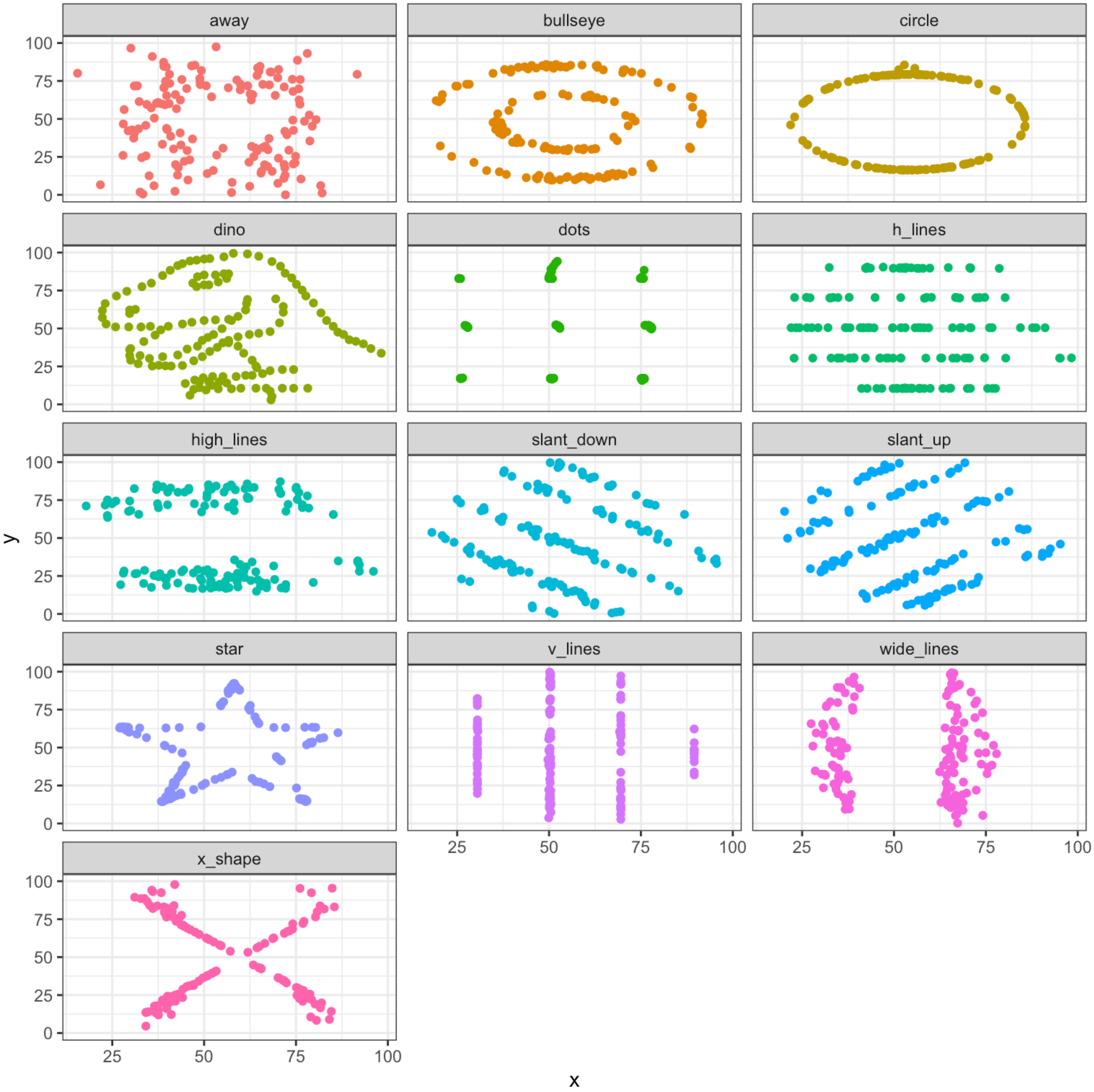
Question 5

```r
par(mar = c(2,2,2,2))
layout(matrix(1:16, nrow=4, ncol=4))
for(name in unique(datasaurus_dozen$dataset)){
  subset <- datasaurus_dozen[datasaurus_dozen$dataset == name, ]
  plot(subset$x, subset$y, main = name)
}
layout(1)
```



```r
# Assign back to default value
par(mar = c(5,4,4,2) + 0.1)
```

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



Question 6

Finally, we want to calculate the correlation between the x and y variables for all 13 datasets. Like before, we will use a loop, but this time, since we want to return a specific value every time through the loop, we will use the sapply function. sapply is useful way to apply a function to every element of a vector. In this case, we provide the vector of unique dataset names (like before) and then our own custom function. This function subsets the data as before, and then returns the correlation coefficient as the output of the function.

```
sapply(unique(df$dataset), function(name){
    subset <- df[df$dataset == name, ]
    return(cor(subset$x, subset$y))
})
```

```
       dino        away     h_lines     v_lines     x_shape        star 
-0.06447185 -0.06412835 -0.06171484 -0.06944557 -0.06558334 -0.06296110 
 high_lines        dots      circle    bullseye    slant_up  slant_down 
-0.06850422 -0.06034144 -0.06834336 -0.06858639 -0.06860921 -0.06897974 
 wide_lines 
-0.06657523 
```