BACS HW Week 9

109090046 assited by 109090035

2023-04-11

Question 1)

Let's make an automated recommendation system for the PicCollage mobile app.

```
library(data.table)

## Warning: 套件 'data.table' 是用 R 版本 4.2.3 來建造的

ac_bundles_dt <- fread("D:/下載/piccollage_accounts_bundles.csv")
ac_bundles_matrix <- as.matrix(ac_bundles_dt[, -1, with=FALSE])
```

a.

Let's explore to see if any sticker bundles seem intuitively similar:

i.

(recommended) Download PicCollage onto your mobile from the App Store and take a look at the style and content of various bundles in their Sticker Store (iOS app: can see how many recommendations does each bundle have? Android app might not have recommendations)

ANS: 6

ii.

Find a single sticker bundle that is both in our limited data set and also in the app's Sticker Store (e.g., "sweetmothersday"). Then, use your intuition to recommend (guess!) five other bundles in our dataset that might have similar usage patterns as this bundle.

ANS: I chose 'betweenspring', and I guess 'saintvalentine', 'HeartStickerPack', 'supersweet', 'togetherwerise' and 'lovestinks2016' might have similar usage patterns as this bundle.

b.

Let's find similar bundles using geometric models of similarity:

i.

Let's create cosine similarity based recommendations for all bundles:

1.

Create a matrix or data.frame of the top 5 recommendations for all bundles

```
library(lsa)

## Warning: 套件 'lsa' 是用 R 版本 4.2.3 來建造的

## 載入需要的套件：SnowballC
```

```
cos_sim_matrix <- round(cosine(ac_bundles_matrix),2)

# Create an empty data frame to store the recommendations
recommendations <- data.frame(bundle = character(),
                              top_1 = character(),
                              top_2 = character(),
                              top_3 = character(),
                              top_4 = character(),
                              top_5 = character(),
                              stringsAsFactors = FALSE)

for (i in 1:nrow(cos_sim_matrix)) {
    top_5_indices <- order(cos_sim_matrix[i,], decreasing = TRUE)[2:6]
    top_5_recommendations <- rownames(cos_sim_matrix)[top_5_indices]
    recommendations <- rbind(recommendations,
                             data.frame(bundle = rownames(cos_sim_matrix)[i],
                                        top_1 = top_5_recommendations[1],
                                        top_2 = top_5_recommendations[2],
                                        top_3 = top_5_recommendations[3],
                                        top_4 = top_5_recommendations[4],
                                        top_5 = top_5_recommendations[5],
                                        stringsAsFactors = FALSE))
}

recommendations %>% head()
```

```
##            bundle            top_1            top_2          top_3      top_4
## 1        Maroon5V       OddAnatomy            alien     beatsmusic       xoxo
## 2         between BlingStickerPack             xoxo           gwen OddAnatomy
## 3      pellington       springrose            8bit2           mmlm julyfourth
## 4     StickerLite HeartStickerPack  HipsterChicSara          Emome    Mom2013
## 5  saintvalentine         nashnext       givethanks togetherwerise  teenwitch
## 6 HipsterChicSara           Random HeartStickerPack     wonderland      Emome
##                    top_5
## 1                   word
## 2 AccessoriesStickerPack
## 3       tropicalparadise
## 4                between
## 5         lovestinks2016
## 6            StickerLite
```

2.

Create a new function that automates the above functionality: it should take an accounts-bundles matrix as a parameter, and return a data object with the top 5 recommendations for each bundle in our data set, using cosine similarity.

```
get_top_5_recommendations <- function(data_matrix) {
  # Compute the cosine similarity matrix
  cos_sim_matrix <- round(cosine(data_matrix),2)

  # Create an empty data frame to store the recommendations
  recommendations <- data.frame(bundle = character(),
                                top_1 = character(),
                                top_2 = character(),
                                top_3 = character(),
```

```
                                top_4 = character(),
                                top_5 = character(),
                                stringsAsFactors = FALSE)

  # Loop through each row (bundle) and get the top 5 recommendations
  for (i in 1:nrow(cos_sim_matrix)) {
    top_5_indices <- order(cos_sim_matrix[i,], decreasing = TRUE)[2:6]
    # Exclude the first index (itself)
    top_5_recommendations <- rownames(cos_sim_matrix)[top_5_indices]
    recommendations <- rbind(recommendations,
                             data.frame(bundle=rownames(cos_sim_matrix)[i],
                                        top_1 = top_5_recommendations[1],
                                        top_2 = top_5_recommendations[2],
                                        top_3 = top_5_recommendations[3],
                                        top_4 = top_5_recommendations[4],
                                        top_5 = top_5_recommendations[5],
                                        stringsAsFactors = FALSE))
  }

  return(recommendations)
}
```

3.

What are the top 5 recommendations for the bundle you chose to explore earlier?

```
top_5 <- get_top_5_recommendations(ac_bundles_matrix)
top_5 <- top_5[top_5$bundle == "betweenspring", ]
top_5
```

```
##           bundle      top_1      top_2   top_3 top_4 top_5
## 35 betweenspring OddAnatomy supersassy between  word   KLL
```

ii.

Let's create correlation based recommendations.

1.

Reuse the function you created above (don't change it; don't use the cor() function)

2.

But this time give the function an accounts-bundles matrix where each bundle (column) has already been mean-centered in advance.

```
# Mean-centered in advance
mean_centered_matrix <- apply(ac_bundles_matrix, 2, function(x) x - mean(x))

top_5_cor <- get_top_5_recommendations(mean_centered_matrix)
```

3.

Now what are the top 5 recommendations for the bundle you chose to explore earlier?

```
top_5_cor <- top_5_cor[top_5_cor$bundle == "betweenspring", ]
top_5_cor
```

```
##           bundle      top_1      top_2   top_3 top_4 top_5
## 35 betweenspring OddAnatomy supersassy between  word   KLL
```

iii.

Let's create adjusted-cosine based recommendations.

1.

Reuse the function you created above (you should not have to change it)

2.

But this time give the function an accounts-bundles matrix where each account (row) has already been mean-centered in advance.

```
adjusted_cosine_matrix <- apply(ac_bundles_matrix, 1, function(x) x - mean(x))

adjusted_cosine_matrix <- t(adjusted_cosine_matrix)
top_5_ad_co <- get_top_5_recommendations(adjusted_cosine_matrix)
```

3.

What are the top 5 recommendations for the bundle you chose to explore earlier?

```
top_5_ad_co <- top_5_ad_co[top_5_ad_co$bundle == "betweenspring", ]
top_5_ad_co

##           bundle      top_1    top_2   top_3 top_4 top_5
## 35 betweenspring OddAnatomy thebouqs between  word  xoxo
```

c.

(not graded) Are the three sets of geometric recommendations similar in nature (theme/keywords) to the recommendations you picked earlier using your intuition alone? What reasons might explain why your computational geometric recommendation models produce different results from your intuition?

ANS: Not similar, I think it's because a bundle may contains lots of theme of stickers.

d.

(not graded) What do you think is the conceptual difference in cosine similarity, correlation, and adjusted-cosine?

ANS: Cosine similarity measures the direction of vectors, correlation measures the linear relationship between variables, and adjusted-cosine similarity measures the relative rating patterns between users. The choice of similarity measure depends on the context and purpose of the analysis, as well as the properties of the data being compared.

Question 2

In our `compstatslib` package, you will find an `interactive_regression()` function that runs a simulation. You can click to add data points to the plotting area and see a corresponding regression line (hitting ESC will stop the simulation). You will also see three numbers: regression intercept – where the regression line crosses the y-axis; regression coefficient – the slope of x on y; correlation - correlation of x and y. For each of the scenarios below, create the described set of points in the simulation. You might

have to create each scenario a few times to get a general sense of them. Visual the scenarios a - d shown below.

```
library(compstatslib)
# interactive_regression()
```

a.

Scenario A: Create a horizontal set of random points, with a relatively narrow but flat distribution.

i.

What raw slope of x and y would you generally expect?

ANS: The raw slope of x and y would be close to zero or very small. This is because the points are distributed horizontally, which means that changes in the x variable will not have a significant impact on the y variable. Therefore, the regression line should be almost flat, indicating a weak relationship between the x and y variables.

ii.

ii. What is the correlation of x and y that you would generally expect?

ANS: The correlation of x and y that you would generally expect would also be close to zero or very small. This is because the correlation coefficient measures the strength of the linear relationship between the x and y variables, and if the points are distributed horizontally, there is little or no linear relationship to measure. Therefore, the correlation coefficient should be close to zero, indicating no or very weak correlation.

b.

Scenario B: Create a random set of points to fill the entire plotting area, along both x-axis and y-axis

i.

What raw slope of the x and y would you generally expect?

ANS: The raw slope of x and y would be non-zero and positive. This is because the points are distributed randomly across the entire plotting area, and some of them are likely to be clustered around the origin or the edges of the plot. Therefore, the regression line should slope upwards from left to right, indicating a positive relationship between the x and y variables.

ii.

What is the correlation of x and y that you would generally expect?

ANS: The correlation of x and y that you would generally expect would be positive or moderately strong. This is because the correlation coefficient measures the strength of the linear relationship between the x and y variables, and if the points are distributed randomly across the entire plot, there is likely to be a positive or moderate linear relationship to measure. Therefore, the correlation coefficient should be positive or moderately strong, indicating a positive or moderate correlation.

c.

Scenario C: Create a diagonal set of random points trending upwards at 45 degrees

i.

What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

ANS: The raw slope of x and y would be close to 1. This is because the points are distributed diagonally, trending upwards at 45 degrees, which means that for every unit increase in the x variable, there will be a corresponding unit increase in the y variable. Therefore, the regression line should slope upwards from left to right at 45 degrees, indicating a strong relationship between the x and y variables.

ii.

What is the correlation of x and y that you would generally expect?

ANS: The correlation of x and y that you would generally expect would be positive and strong. This is because the correlation coefficient measures the strength of the linear relationship between the x and y variables, and if the points are distributed diagonally, trending upwards at 45 degrees, there is likely to be a strong linear relationship to measure. Therefore, the correlation coefficient should be positive and strong, indicating a strong correlation.

d.

Scenario D: Create a diagonal set of random trending downwards at 45 degrees

i.

What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

ANS: The raw slope of x and y would be close to -1. This is because the points are distributed diagonally, trending downwards at 45 degrees, which means that for every unit increase in the x variable, there will be a corresponding unit decrease in the y variable. Therefore, the regression line should slope downwards from left to right at 45 degrees, indicating a strong relationship between the x and y variables.

ii.

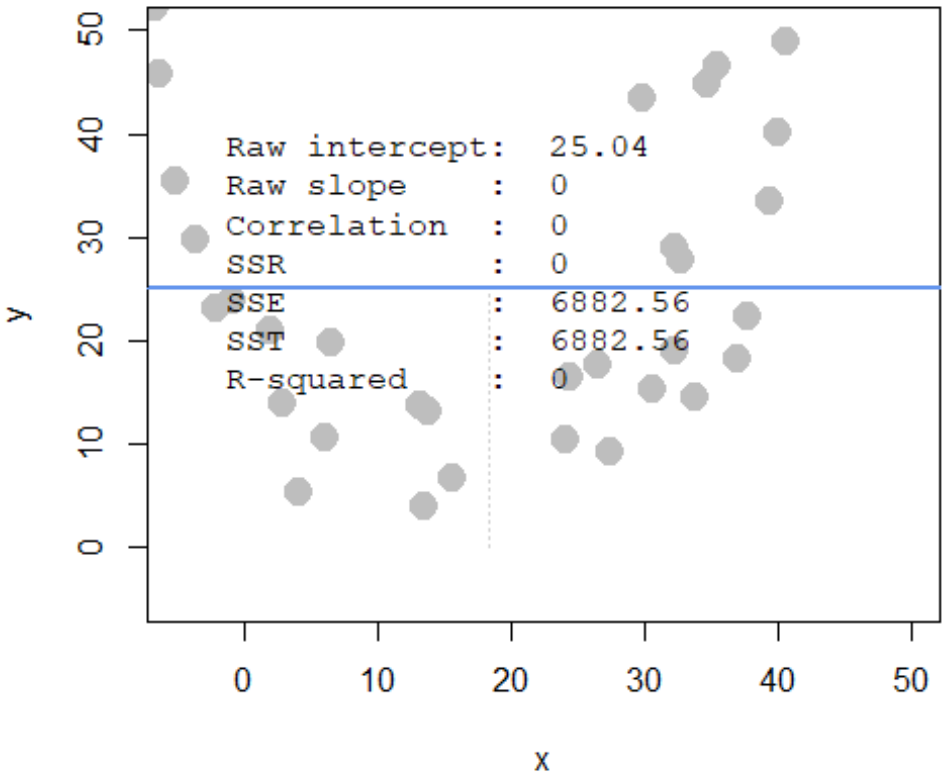
What is the correlation of x and y that you would generally expect?

ANS: The correlation of x and y that you would generally expect would be negative and strong. This is because the correlation coefficient measures the strength of the linear relationship between the x and y variables, and if the points are distributed diagonally, trending downwards at 45 degrees, there is likely to be a strong linear relationship to measure. Therefore, the correlation coefficient should be negative and strong, indicating a strong negative correlation.

e.

Apart from any of the above scenarios, find another pattern of data points with no correlation ($r \approx 0$). (can create a pattern that visually suggests a strong relationship but produces $r \approx 0$?)

U-shaped distribution

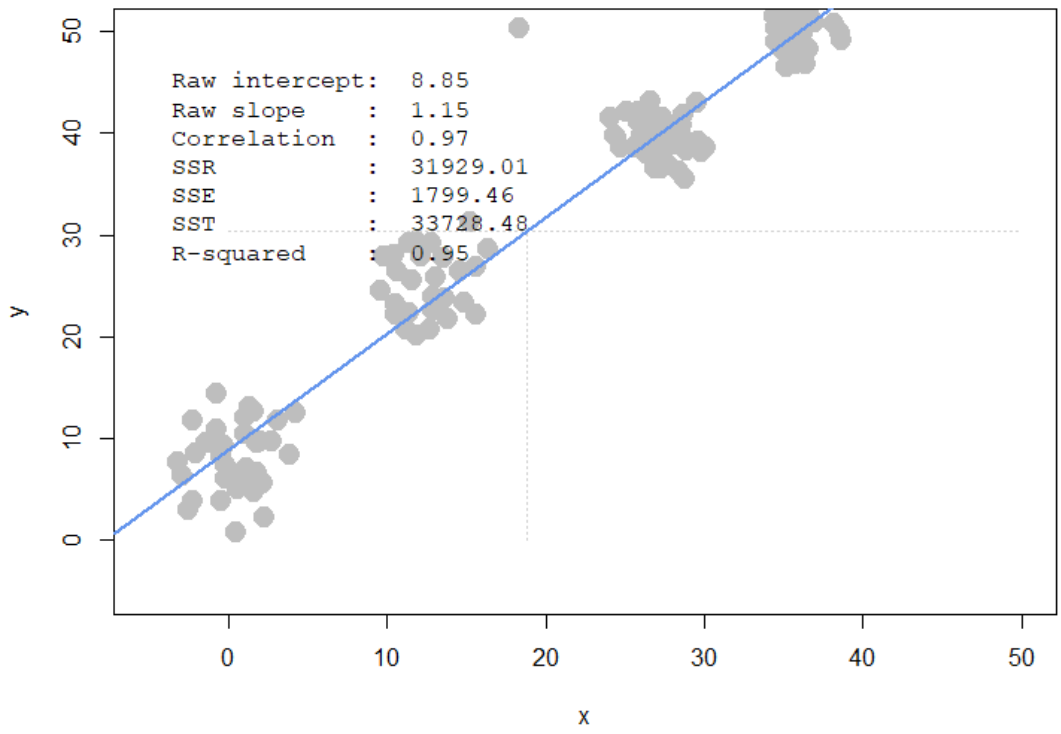


Visually, this pattern can suggest a strong non-linear relationship between the x and y variables, with the majority of the points clustered at the extremes of both variables. However, because the points are distributed symmetrically around the origin, there is no significant linear relationship to measure, and the correlation coefficient will be close to zero.

f.

Apart from any of the above scenarios, find another pattern of data points with perfect correlation ($r \approx 1$). (can you find a scenario where the pattern visually suggests a different relationship?)

a set of data points that are clustered in several groups, with each group forming a distinct "step" along one of the variables.



g.

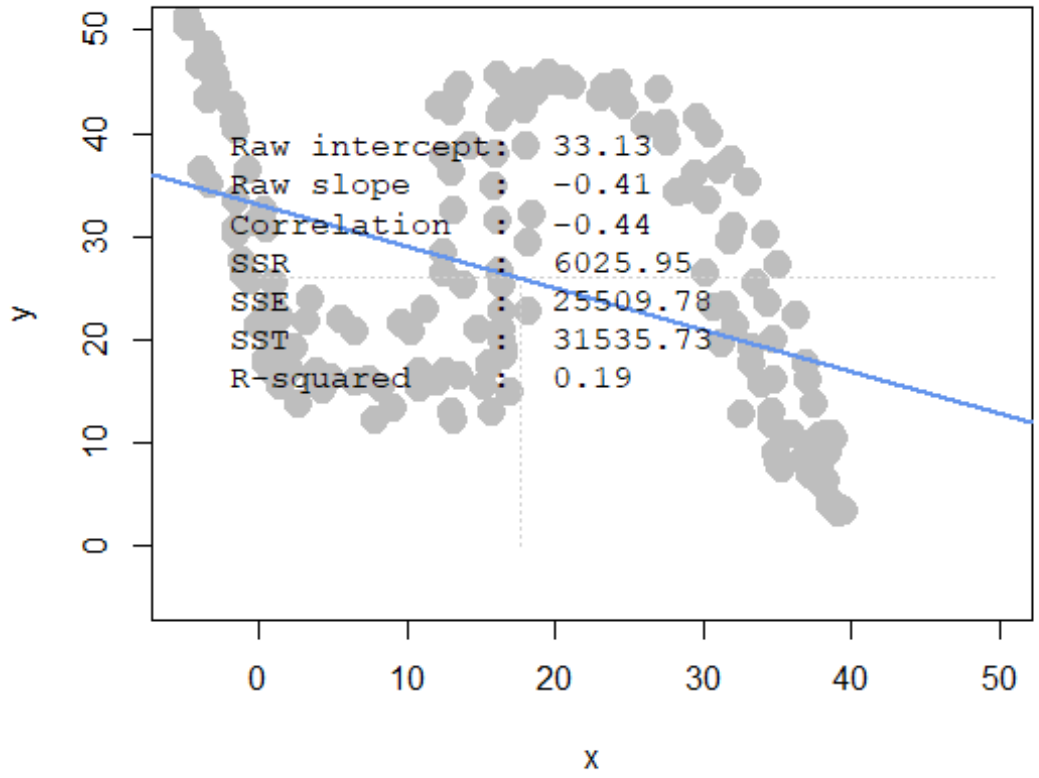
Let's see how correlation relates to simple regression, by simulating any linear relationship you wish:

i.

Run the simulation and record the points you create: `pts <- interactive_regression()` (simulate either a positive or negative relationship)

```
# pts <- interactive_regression()
library(readr)

## Warning: 套件 'readr' 是用 R 版本 4.2.2 來建造的

pts <- read_csv("D:/下載/interactive_regression_result.csv")

## Rows: 130 Columns: 2
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## dbl (2): x, y
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.

pts

## # A tibble: 130 × 2
##         x      y
##     <dbl>  <dbl>
##  1 -3.17   7.73
##  2 -2.56   2.93
```

```
##  3  0.522  0.783
##  4  2.25   2.25
##  5  3.91   8.41
##  6  4.29  12.5
##  7 -0.757 14.4
##  8 -2.26  11.7
##  9 -0.757 10.9
## 10  1.05  12.0
## # … with 120 more rows
```



ii.

Use the `lm()` function to estimate the regression intercept and slope of pts to ensure they are the same as the values reported in the simulation plot: `summary( lm( pts$y ~ pts$x ))`

```
lm_results <- summary(lm(pts$y ~ pts$x))
intercept <- lm_results$coefficients[1,1]
slope <- lm_results$coefficients[2,1]

## The regression intercept is 8.85

## The regression slope is 1.15
```

The results are same as the values reported in the simulation plot.

iii.

Estimate the correlation of x and y to see it is the same as reported in the plot: `cor(pts)`

```
cor(pts)

##           x         y
## x 1.0000000 0.9729587
## y 0.9729587 1.0000000

## The correlation is 0.97
```

The result is same as the value reported in the simulation plot.

iv.

Now, standardize the values of both x and y from pts and re-estimate the regression slope

```
std_x <- scale(pts$x)
std_y <- scale(pts$y)
std_slope <- cor(std_x, std_y) * sd(std_y) / sd(std_x)

## The re-estimated the regression slope is 0.97
```

v.

What is the relationship between correlation and the standardized simple-regression estimates?

ANS:

The relationship between correlation and the standardized simple-regression estimates can be described by the formula: *standardized slope = correlation * (standard deviation of y) / (standard deviation of x).*

This means that the standardized slope (the slope of the regression line when both x and y are standardized) is equal to the correlation coefficient multiplied by the ratio of the standard deviation of y to the standard deviation of x. In other words, *the stronger the correlation, the larger the standardized slope, and the weaker the correlation, the smaller the standardized slope.*