BACS - HW9

109006241

Question 1) Let's make an automated recommendation system for the PicCollage mobile app.

Download the CSV file piccollage_accounts_bundles.csv from Canvas.

You may either use read.csv() and data.frame to load the file as before, or you can optionally try learning how to use data.table – a high performance package for reading, writing, and managing large data sets.

```
ac_bundles_dt = fread("D:/Users/User/Documents/R/BACS/Homeworks/HW9/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)

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.

> I picked the Maroon 5 V sticker bundle. Based on my intuition, the 5 other bundles that I recommend are: "Funky Geometry", "Pink Balloons", "Futuristic Shapes", "Diamonds & Pearls", and "Thinking of You"

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

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

```
cos_sim = cosine(ac_bundles_matrix)
```

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

```
get_top5 = function(x) {
  return (names(sort(x, decreasing=T))[2:6])
}

top5_all = t(apply(cos_sim, 1, get_top5))
head(top5_all, 5)
```

```
##                [,1]               [,2]              [,3]
## Maroon5V       "OddAnatomy"       "beatsmusic"      "xoxo"
## between        "BlingStickerPack" "xoxo"            "gwen"
## pellington     "springrose"       "8bit2"           "mmlm"
## StickerLite    "HeartStickerPack" "HipsterChicSara" "Mom2013"
## saintvalentine "nashnext"         "givethanks"      "teenwitch"
##                [,4]             [,5]
## Maroon5V       "alien"          "word"
## between        "OddAnatomy"     "AccessoriesStickerPack"
## pellington     "julyfourth"     "tropicalparadise"
## StickerLite    "Emome"          "Random"
## saintvalentine "togetherwerise" "lovestinks2016"
```

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.

```
top5_rec = function(x, y) {
  cos_sim = cosine(x)
  top5 = get_top5(cos_sim[y,])
  return (top5)
}
```

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

```
top5_rec(ac_bundles_matrix, "Maroon5V")
```

```
## [1] "OddAnatomy" "beatsmusic" "xoxo"       "alien"      "word"
```

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.

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

```
bundle_means = apply(ac_bundles_matrix, 2, mean)
bundle_means_matrix = t(replicate(nrow(ac_bundles_matrix), bundle_means))
ac_bundles_mc_b = ac_bundles_matrix - bundle_means_matrix
top5_rec(ac_bundles_mc_b, "Maroon5V")
```

```
## [1] "OddAnatomy" "beatsmusic" "xoxo"       "alien"      "word"
```

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.

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

```r
bundle_means = apply(ac_bundles_matrix, 1, mean)
bundle_means_matrix = replicate(ncol(ac_bundles_matrix), bundle_means)
ac_bundles_mc_b = ac_bundles_matrix - bundle_means_matrix
top5_rec(ac_bundles_mc_b, "Maroon5V")
```

```
## [1] "OddAnatomy" "word"       "xoxo"       "beatsmusic" "supercute"
```

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?

> No, it is not. Maybe it's because our intuition is just based on visual intuition, while the three geometric recommendation methods are based on a more mathemathical approach.

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

> Cosine similarity uses the cosine of the angle between the two vectors being compared, correlation measures their linear relationship, and adjusted-cosine is like cosine similarity, but it also takes into account the some biases, such as rating biases of users.

Question 2) Correlation is at the heart of many data analytic methods so let's explore it further.

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.

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?

> It will be very close to 0, because if the data forms a straight horizontal line, the slope will be close to 0.

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

> It will be very close to 0 as well, because the values on the x-axis don't have a significant impact towards the values on the y-axis.

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?

It will be very close to 0, because if the data is just randomly distributted, filling the entire plotting area equally, the slope will be close to 0.

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

It will be very close to 0 as well, because the values on the x-axis don't have a significant impact towards the values on the y-axis. They have a very little correlation because the data is too spread out.

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)

It will be close to 1, because as we can see, a 1 unit increase in the x-axis will also result in a 1 unit increase in the y-axis.

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

It will be close to 1, because they have a strong positive correlation.

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

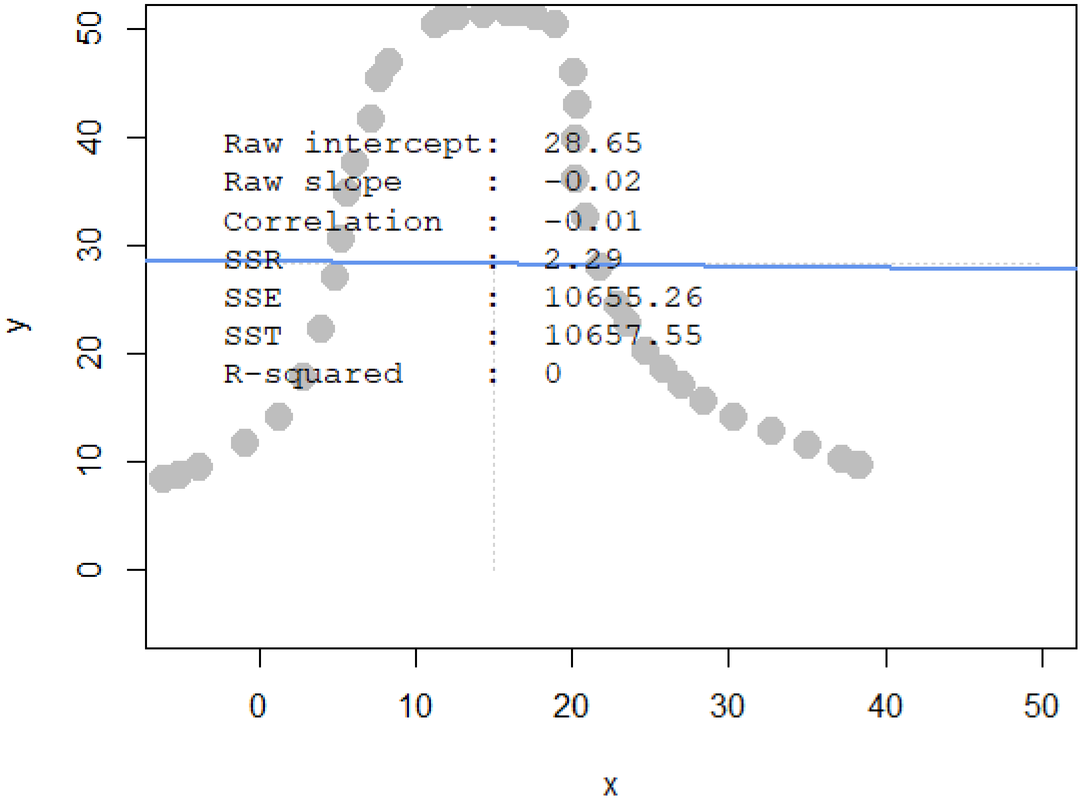
i. What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

It will be close to -1, because as we can see, a 1 unit increase in the x-axis will result in a -1 unit increase in the y-axis.

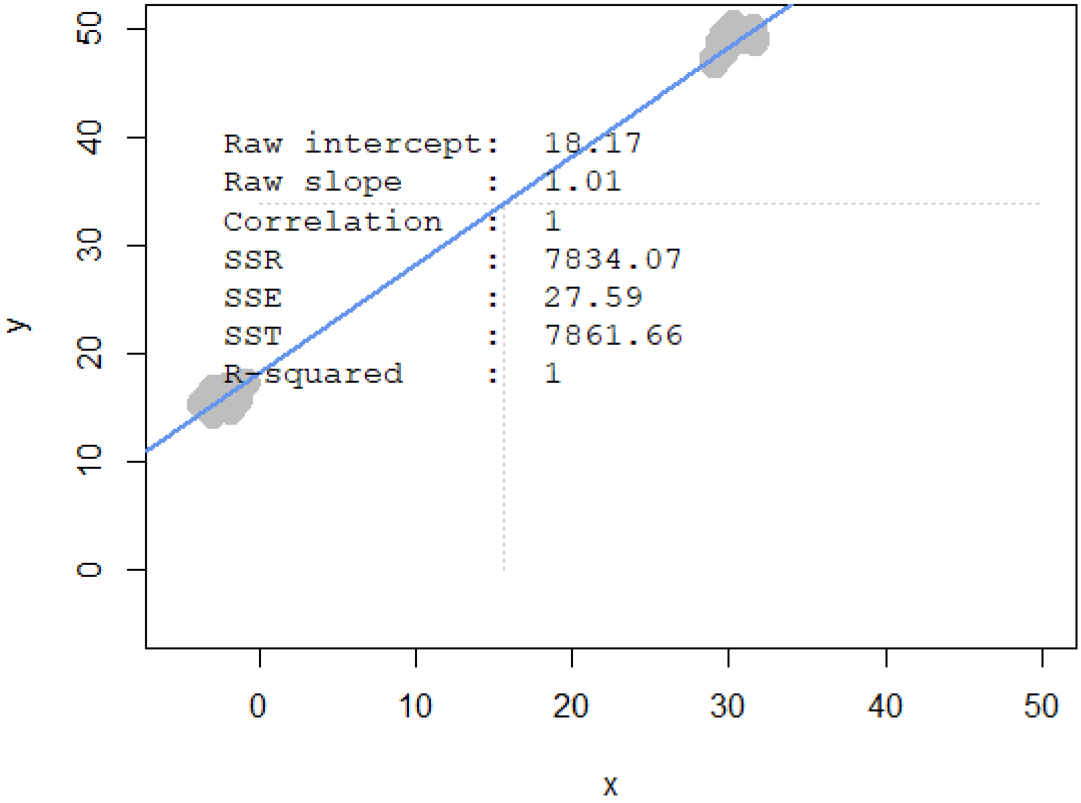
ii. What is the correlation of x and y that you would generally expect?

It will be close to -1, because they have a strong negative correlation.

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



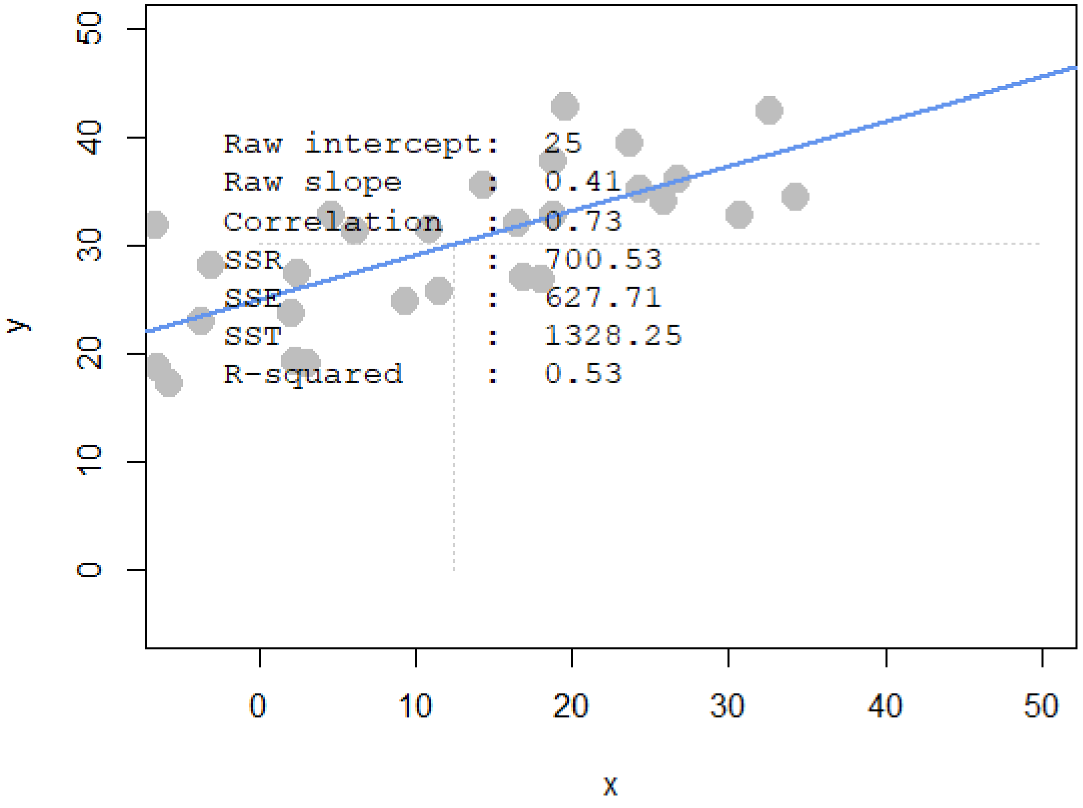
f. Apart from any of the above scenarios, find another pattern of data points with perfect correlation (r = 1). (can you find a scenario where the pattern visually suggests a different relationship?)



```
Raw intercept:  18.17
Raw slope     :  1.01
Correlation   :  1
SSR           :  7834.07
SSE           :  27.59
SST           :  7861.66
R-squared     :  1
```

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)



```
# The 'regression_pts.csv' file contains points produced by running the interactive_regression()
# in the terminal, which has been saved as a csv file before.
pts = read.csv("D:/Users/User/Documents/R/BACS/Homeworks/HW9/regression_pts.csv")
```

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))

```
summary( lm( pts$y ~ pts$x ))
```

```
##
## Call:
## lm(formula = pts$y ~ pts$x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3032 -4.2377  0.0517  4.1257  9.6683
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25.00400    1.33144  18.780  < 2e-16 ***
## pts$x        0.41184    0.07645   5.387 1.21e-05 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 4.914 on 26 degrees of freedom
## Multiple R-squared:  0.5274, Adjusted R-squared:  0.5092
## F-statistic: 29.02 on 1 and 26 DF,  p-value: 1.215e-05
```

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 0.7262311
## y 0.7262311 1.0000000
```

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

```
pts_std = pts
pts_std$x = (pts$x - mean(pts$x)) / sd(pts$x)
pts_std$y = (pts$y - mean(pts$y)) / sd(pts$y)
summary( lm( pts_std$y ~ pts_std$x ))
```

```
##
## Call:
## lm(formula = pts_std$y ~ pts_std$x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.04125 -0.60419  0.00737  0.58821  1.37846
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.328e-16  1.324e-01   0.000        1
## pts_std$x    7.262e-01  1.348e-01   5.387 1.21e-05 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.7005 on 26 degrees of freedom
## Multiple R-squared:  0.5274, Adjusted R-squared:  0.5092
## F-statistic: 29.02 on 1 and 26 DF,  p-value: 1.215e-05
```

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

The correlation is equal to the standardized simple-regression slope estimate.