106073401

BACS HW (Week 7)

1. a) i. There are 6 recommendations each bundle have.

   ii. If I click into “CutieV” bundles, I guess “creppycute”, “hellobaby”, “CatpixCubie”, “babyanimals”,”AnimalsFriendsStickerPack” might also be recommended.

   b) 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(data.table)
ac_bundles_dt = fread("piccollage_accounts_bundles.csv")
ac_bundles_matrix = as.matrix(ac_bundles_dt[, -1, with=FALSE])
library(lsa)
cosine_similarity <- cosine(ac_bundles_matrix)
top5 <- t(apply(cosine_similarity,1,function(x) names(sort(x,decreasing = TRU
E))))[,2:6]
```

   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.

```
recommendation <- function(matrix)
{
  cosine_similarity <- cosine(matrix)
  top5 <- t(apply(cosine_similarity,1,function(x) names(sort(x,decreasing = TRU
E))))[,2:6]
  return(top5)
}
```

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

```
recommendation(ac_bundles_matrix)["CutieV",]
[1] "Valentine2013StickerPack" "AntiV"                    "Mom2013"
[4] "wonderland"               "Random"
```

   ii. Let’s create *correlation* based recommendations.

```
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
recommendation(ac_bundles_mc_b)["CutieV",]
[1] "Valentine2013StickerPack" "AntiV"                    "Mom2013"
[4] "wonderland"               "Random"
```

iii. Let's create adjusted-cosine based recommendations..

```
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
recommendation(ac_bundles_mc_b)["CutieV",]
[1] "Valentine2013StickerPack" "AntiV"                    "Mom2013"
[4] "wonderland"               "Eggotown"
```

106073401

2. 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?

      0

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

      0

   b) Create a completely 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?

      0

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

      0

   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)

      1

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

      1

   d) Create a diagonal set of random trending downwards at 45 degrees.

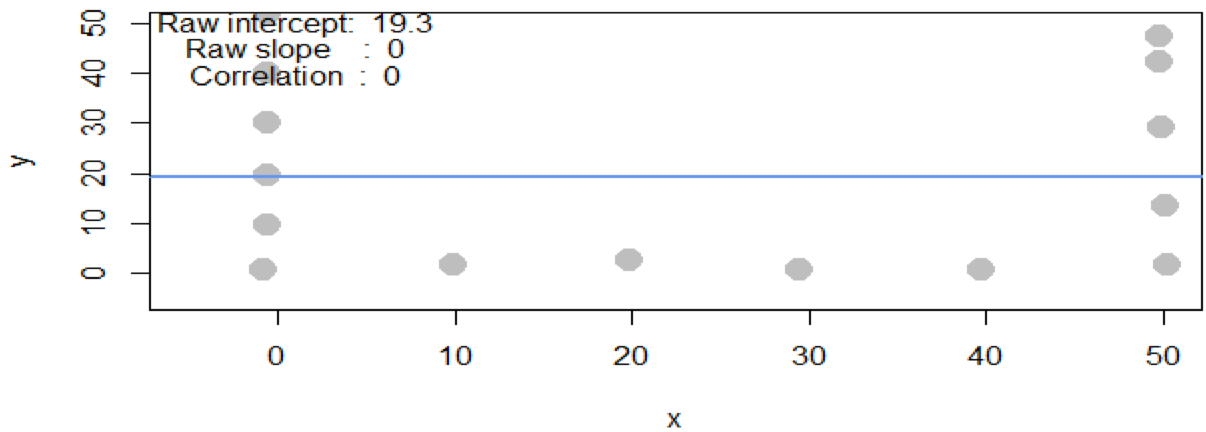
   i. What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

      -1

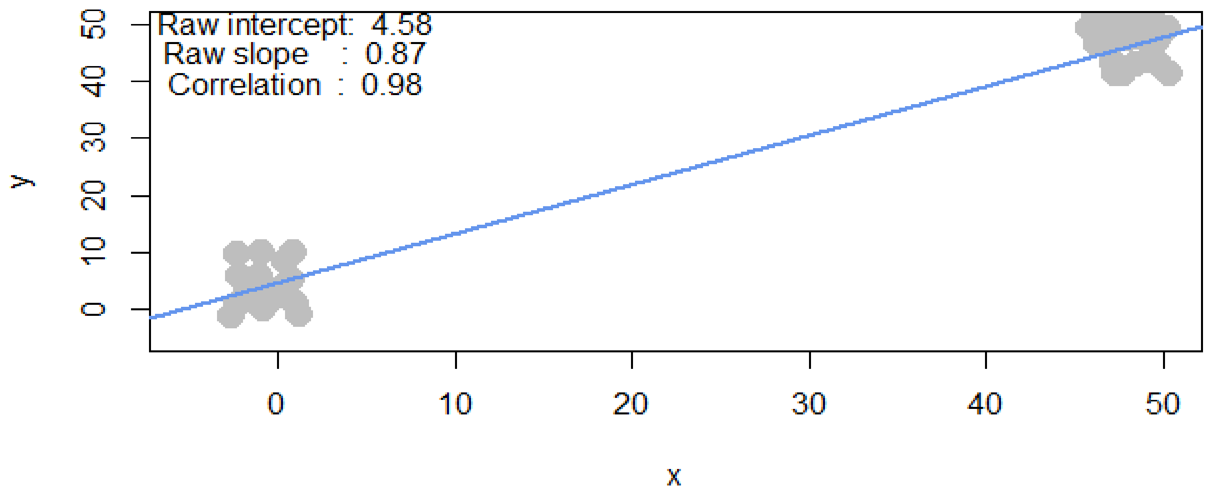
   ii. What is the correlation of x and y that you would generally expect?

      -1

   e) Apart from any of the above scenarios, find another pattern of data points with no correlation (r ≈ 0).



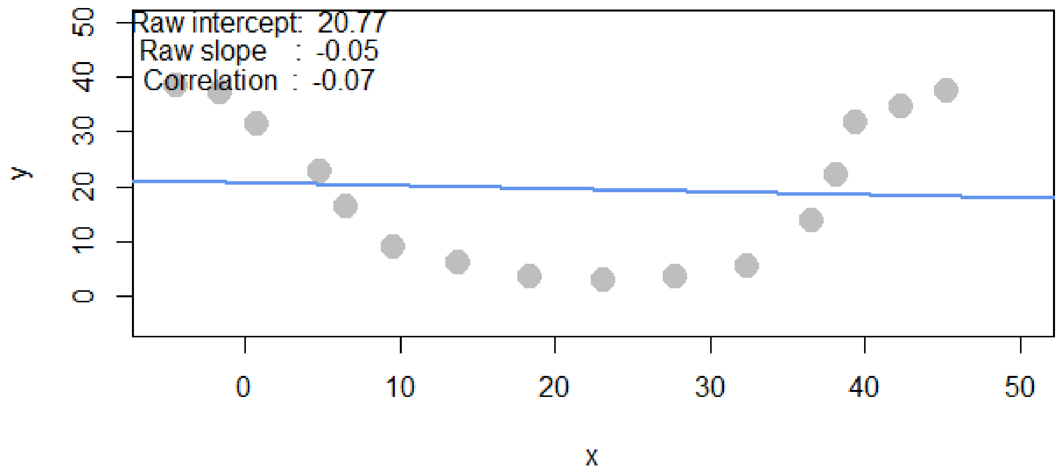
   f) Apart from any of the above scenarios, find another pattern of data points with perfect correlation (r ≈ 1)



106073401

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()



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 
-16.759 -13.787  -1.035  13.720  19.044 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  20.7696     5.7517   3.611  0.00284 **
pts$x        -0.0539     0.2164  -0.249  0.80691   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14.23 on 14 degrees of freedom
Multiple R-squared:  0.004412, Adjusted R-squared:  -0.0667 
F-statistic: 0.06204 on 1 and 14 DF,  p-value: 0.8069
```

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  1.00000000 -0.06642432
y -0.06642432  1.00000000
```

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

```
pts_std <- data.frame(scale(pts))
summary( lm( pts_std$y ~ pts_std$x ))
Call:
lm(formula = pts_std$y ~ pts_std$x)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.21613 -1.00046 -0.07513  0.99562  1.38198 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.551e-17  2.582e-01   0.000    1.000
pts_std$x   -6.642e-02  2.667e-01  -0.249    0.807

Residual standard error: 1.033 on 14 degrees of freedom
Multiple R-squared:  0.004412,	Adjusted R-squared:  -0.0667 
F-statistic: 0.06204 on 1 and 14 DF,  p-value: 0.8069
```

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

If the correlation almost equal to 0, the R-square will be smaller..