423 project

Chris Chen

1/20/2022

Preliminaries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(expm)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
##
## Attaching package: 'expm'
```

```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
library(ggplot2)
```

Dataset

```
perfume = read.csv("noon_perfumes_dataset.csv")
sum(is.na(perfume))
```

```
## [1] 0
```

```
head(perfume)
```

```
##   X           brand              name old_price new_price  ml concentration
```

```
## 1 0    PACO RABANNE   1 Million Lucky       395    244.55 100          EDT
## 2 1 Roberto Cavalli Paradiso Assoluto       415    107.95  50          EDP
## 3 2      S.T.Dupont       Royal Amber       265    186.90 100          EDP
## 4 3           GUESS    Seductive Blue       290    103.20 100          EDT
## 5 4 Roberto Cavalli              Uomo       260     94.95  50          EDP
## 6 5 Roberto Cavalli           cavalli       260     94.95  50          EDP
##   department  scents                                    base_note
## 1        Men   Woody              Oakmoss, Patchouli and Vetiver
## 2      Women  Floral           Vanilla, Sandalwood And Patchouli
## 3     Unisex Arabian                    Lemon, Mint and Wood Moss
## 4        Men   Spicy Cashmere Wood, Moss And Rippled Sand Accord
## 5      Women Arabian                Vanille, Benzoin, Tonka Bean
## 6      Women Arabian                Vanille, Benzoin, Tonka Bean
##                                       middle_note item_rating seller
## 1 Hazelnut, Jasmine, Cashmir Wood, Cedar and Honey         5.0   noon
## 2                         Wild Jasmine and Red Lily         4.8   noon
## 3                              Sandalwood and Cedar         5.0   noon
## 4         Blue Coral Aquaspace Accord And Geranium         3.0   noon
## 5                             African Orange Flower         4.8   noon
## 6                             African Orange Flower         4.8   noon
##   seller_rating num_seller_ratings
## 1           4.2              98.1K
## 2           4.2              98.1K
## 3           4.2              98.1K
## 4           4.2              98.1K
## 5           4.2              98.1K
## 6           4.2              98.1K
```

no empty value. good.

```
perfume = perfume %>%
  mutate(scent = ifelse(scents == "Arabian", "Oriental", scents))
p1 = subset(perfume, scent != "Vanilla" & scent != "Aromatic" & scent != "Musk" & scent != "Jasmine" & s
```

```
p2 = p1 %>%
  mutate(conc = ifelse(concentration == "PDT", "EDT", concentration))
p2 = subset(p2, select = -c(concentration))
```

```
p3 = p2 %>%
  mutate(brands1 = ifelse(brand == "ST Dupont", "S.T.Dupont", brand)) %>%
  mutate(brands2 = ifelse(brands1 == "armani", "GIORGIO ARMANI", brands1)) %>%
  mutate(brands3 = ifelse(brands2 == "Genie Collection", "Genie", brands2)) %>%
  mutate(brands4 = ifelse(brands3 == "LANVIN PARIS", "LANVIN", brands3)) %>%
  mutate(brands5 = ifelse(brands4 == "Mont Blanc", "MONTBLANC", brands4)) %>%
  mutate(brands6 = ifelse(brands5 == "marbert man", "Marbert", brands5)) %>%
  mutate(brands = ifelse(brands6 == "YSL" | brands6 == "YVES", "Yves Saint Laurent", brands6))
p3 = subset(p3, select = -c(brand, brands1, brands2, brands3, brands4, brands5, brands6))
```

```
p4 = subset(p3, seller_rating <= 5.0)
p5 = p4 %>%
  mutate(num_sel_ratings =
           ifelse(grepl("K", num_seller_ratings),
                  as.numeric(substring(num_seller_ratings, 1, nchar(num_seller_ratings) - 1)) * 1000,
                  as.numeric(num_seller_ratings)))
```

```
## Warning in ifelse(grepl("K", num_seller_ratings),
```

```
## as.numeric(substring(num_seller_ratings, : NAs introduced by coercion
```

```
p5 = subset(p5, select = -c(num_seller_ratings))
```

```
# clean seller column
seller = as.vector(p5$seller)
seller = tolower(seller)
index_golden = which(grepl("golden", seller))
seller[index_golden] = "golden perfumes"
index_lolita = which(grepl("lolita", seller))
seller[index_lolita] = "lolita shop"
index_noon = which(grepl("noon", seller))
seller[index_noon] = "noon"
index_swiss = which(grepl("swiss", seller))
seller[index_swiss] = "swiss arabian perfumes"
index_pa = which(grepl("perfumes--addresses", seller))
seller[index_pa] = "perfumes"
index_ps = which(grepl("perfumes-shop", seller))
seller[index_ps] = "perfumes"

p6 = p5
p6$seller = seller
sb = c(48, 435, 651)
bf = c(109, 121, 470, 565, 576)
p6 = p6 %>%
  mutate(seller1 = ifelse(is.element(X, sb), "show biz", seller)) %>%
  mutate(sellers = ifelse(is.element(X, bf), "beauty fortune", seller))
p6 = subset(p6, select = -c(seller1, seller))
```

```
base_note = as.vector(p6$base_note)
base_note = tolower(base_note)
base_note = str_replace_all(base_note, " and ", ",")
base_note = str_replace_all(base_note, " ", "")
base_note = str_replace_all(base_note, "vanille", "vanilla")
base_note = str_replace_all(base_note, "woodsynotes", "wood")
base_note = str_replace_all(base_note, "orrisroot", "orris")
base_note = str_replace_all(base_note, "woodsynote", "wood")
base_note = str_replace_all(base_note, "woodynotes", "wood")
base_note = str_replace_all(base_note, "woody", "wood")
base_note = str_replace_all(base_note, "cedarwood", "cedar")
base_note = str_replace_all(base_note, "virginiacedar", "cedar")
base_note = str_replace_all(base_note, "whitemusk", "musk")
base_note = str_replace_all(base_note, "tonkabeans", "tonka")
base_note = str_replace_all(base_note, "tonkabean", "tonka")
base_note = str_replace_all(base_note, "amberwood", "amber")
base_note = str_replace_all(base_note, "sandalwood", "sandal")
base_note = str_replace_all(base_note, "cashmerewood", "cashmere")
base_note = str_replace_all(base_note, "guaiacwood", "guaiac")
base_note = str_replace_all(base_note, "ambergris", "AMBERGRIS")
base_note = str_replace_all(base_note, "mustyoud", "oud")
base_note = str_replace_all(base_note, "naturaloudoil", "oud")
base_note = str_replace_all(base_note, "agarwood\\(oud\\)", "oud")
base_note = str_replace_all(base_note, "agarwood", "oud")
base_note = str_replace_all(base_note, "oudh", "oud")
p6$base_note = base_note
```

```
mid_note = as.vector(p6$middle_note)
mid_note = tolower(mid_note)
mid_note = str_replace_all(mid_note, " and ", ",")
mid_note = str_replace_all(mid_note, " ", "")
mid_note = str_replace_all(mid_note, "lily-of-the-valley", "lily")
mid_note = str_replace_all(mid_note, "orrisroot", "orris")
mid_note = str_replace_all(mid_note, "lilyofthevalley", "lily")
mid_note = str_replace_all(mid_note, "bulgarianrose", "rose")
mid_note = str_replace_all(mid_note, "africanorangeflower", "orangeblossom")
mid_note = str_replace_all(mid_note, "neroli", "orangeblossom")
mid_note = str_replace_all(mid_note, "jasminesambac", "jasmine")
mid_note = str_replace_all(mid_note, "wildjasmine", "jasmine")
mid_note = str_replace_all(mid_note, "wildjasmine", "jasmine")
mid_note = str_replace_all(mid_note, "blackpepper", "pepper")
mid_note = str_replace_all(mid_note, "pinkpepper", "pepper")
mid_note = str_replace_all(mid_note, "vanille", "vanilla")
mid_note = str_replace_all(mid_note, "tuberose", "TUBEROSE")
mid_note = str_replace_all(mid_note, "orrisroot", "ORRISROOT")
mid_note = str_replace_all(mid_note, "honeysuckle", "HONEYSUCKLE")
mid_note = str_replace_all(mid_note, "rosemary", "ROSEMARY")
mid_note = str_replace_all(mid_note, "violetleaf", "VIOLETLEAF")
mid_note = str_replace_all(mid_note, "clarysage", "CLARYSAGE")
mid_note = str_replace_all(mid_note, "oudh", "oud")
mid_note = str_replace_all(mid_note, "burningoud", "oud")
mid_note = str_replace_all(mid_note, "agarwood\\(oud\\)", "oud")
mid_note = str_replace_all(mid_note, "agarwood", "oud")
mid_note = str_replace_all(mid_note, "oudwood", "oud")
p6$middle_note = mid_note
```

```
# clean ml column
vol = as.vector(p6$ml)
del_vol = as.data.frame(vol) %>%
  group_by(vol) %>%
  summarise(count = n()) %>%
  filter(count <= 5) %>%
  subset(select = vol)
del_vol = as.vector(del_vol$vol)

p7 = p6
index_del = which(p7$ml %in% del_vol)
p7 = p7[-index_del, ]

# add ordinal version of ml
vol = as.vector(p7$ml)
unique_vol = as.data.frame(vol) %>%
  group_by(vol) %>%
  summarise(count = n()) %>%
  subset(select = vol)
unique_vol = as.vector(unique_vol$vol)

order = vol
rank = 0
for (i in unique_vol) {
  rank = rank + 1
```

```
  index = which(vol == i)
  order[index] = rank
}
p7$ml_order = order
p7 = subset(p7, select = -c(ml))
```

```
perfume = subset(p7, select = -c(X, name, scents))
perfume = unique(perfume)

brand = as.vector(p7$brands)
brand = tolower(brand)
new_brands = as.data.frame(brand) %>%
  group_by(brand) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
big_brands = new_brands[which(new_brands$count > 10), ]$brand
perfume = perfume %>%
  mutate(big_brand = ifelse(is.element(tolower(brands), big_brands), 1, 0))
perfume = subset(perfume, select = -c(brands))

perfume = perfume %>%
  mutate(is_noon = ifelse(tolower(sellers) == 'noon', 1, 0))
perfume = subset(perfume, select = -c(sellers))

get_notes = function(base, middle) {
  bnote = as.vector(unlist(strsplit(base, split = ",")))
  mnote = as.vector(unlist(strsplit(middle, split = ",")))
  return(union(bnote, mnote))
}

complexity = function(notes) {
  return(length(notes))
}

luxury = function(notes) {
  score = 0
  for (i in 1:length(notes)) {
    if (notes[i] == "musk" | notes[i] == "orris") {
      score = score + 1
    } else if (notes[i] == "neroli" | notes[i] == "jasmine" | notes[i] == "sandal") {
      score = score + 2
    } else if (notes[i] == "rose" | notes[i] == "tuberose") {
      score = score + 3
    } else if (notes[i] == "AMBERGRIS") {
      score = score + 4
    } else if (notes[i] == "oud") {
      score = score + 5
    } else {
      score = score + 0
    }
  }
  return(score)
}
```

```r
N = nrow(perfume)
complex = lux = rep(0, N)
for (i in 1:N) {
  complex[i] = complexity(get_notes(perfume[i, ]$base_note, perfume[i, ]$middle_note))
  lux[i] = luxury(get_notes(perfume[i, ]$base_note, perfume[i, ]$middle_note))
}
comp_score = lux_score = rep(0, N)
for (i in 1:N) {
  x = complex[i]
  comp_score[i] = sum(complex <= x) / N * 100
  y = lux[i]
  lux_score[i] = sum(lux <= y) / N * 100
}
nose_score = comp_score * lux_score / 100
perfume = perfume %>%
  mutate(nose_rating = nose_score)
```

```r
lm.1 = lm(old_price ~ big_brand + is_noon + nose_rating + item_rating +
            department + conc + ml_order +
            seller_rating + scent + num_sel_ratings, data = perfume)
summary(lm.1)
```

```
##
## Call:
## lm(formula = old_price ~ big_brand + is_noon + nose_rating +
##     item_rating + department + conc + ml_order + seller_rating +
##     scent + num_sel_ratings, data = perfume)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -397.23 -146.26  -17.12  115.35 1927.79
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -5.236e+02  3.607e+02  -1.452   0.1470
## big_brand         7.519e+01  1.678e+01   4.482 8.53e-06 ***
## is_noon           9.172e+01  9.847e+01   0.931   0.3519
## nose_rating      -7.140e-01  3.078e-01  -2.320   0.0206 *
## item_rating       7.597e+00  1.451e+01   0.523   0.6008
## departmentMen     2.894e+02  2.304e+02   1.256   0.2095
## departmentUnisex  1.829e+02  2.321e+02   0.788   0.4310
## departmentWomen   2.768e+02  2.292e+02   1.207   0.2276
## concEDP           2.241e+02  2.296e+02   0.976   0.3294
## concEDT           6.419e+01  2.290e+02   0.280   0.7793
## ml_order          1.569e+01  3.662e+00   4.286 2.05e-05 ***
## seller_rating     6.794e+01  4.147e+01   1.638   0.1018
## scentFloral      -8.053e+00  3.169e+01  -0.254   0.7995
## scentFresh       -9.492e+01  4.476e+01  -2.121   0.0342 *
## scentFruity      -5.218e+01  3.955e+01  -1.319   0.1874
## scentOriental    -6.088e+01  3.746e+01  -1.625   0.1046
## scentSpicy       -3.624e+01  3.492e+01  -1.038   0.2997
## scentWoody        1.735e+01  3.203e+01   0.542   0.5882
## num_sel_ratings  -1.271e-03  1.017e-03  -1.250   0.2116
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 227.5 on 777 degrees of freedom
## Multiple R-squared:  0.1439, Adjusted R-squared:  0.1241
## F-statistic: 7.256 on 18 and 777 DF,  p-value: < 2.2e-16
```

```
lm.2 = lm(item_rating ~ big_brand + is_noon + nose_rating + old_price +
            department + conc + ml_order +
            seller_rating + scent + num_sel_ratings, data = perfume)
summary(lm.2)
```

```
##
## Call:
## lm(formula = item_rating ~ big_brand + is_noon + nose_rating +
##     old_price + department + conc + ml_order + seller_rating +
##     scent + num_sel_ratings, data = perfume)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4686 -0.1553  0.1002  0.3573  0.7068
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.815e+00  8.819e-01   4.325 1.72e-05 ***
## big_brand         1.788e-02  4.199e-02   0.426   0.6704
## is_noon           2.207e-02  2.435e-01   0.091   0.9278
## nose_rating      -5.253e-04  7.630e-04  -0.688   0.4914
## old_price         4.640e-05  8.864e-05   0.523   0.6008
## departmentMen    -4.798e-01  5.698e-01  -0.842   0.4000
## departmentUnisex -5.048e-01  5.736e-01  -0.880   0.3791
## departmentWomen  -3.706e-01  5.668e-01  -0.654   0.5134
## concEDP           4.886e-01  5.674e-01   0.861   0.3894
## concEDT           3.883e-01  5.658e-01   0.686   0.4928
## ml_order         -3.813e-03  9.154e-03  -0.417   0.6772
## seller_rating     1.977e-01  1.024e-01   1.931   0.0539 .
## scentFloral      -7.436e-02  7.826e-02  -0.950   0.3423
## scentFresh       -1.251e-01  1.108e-01  -1.129   0.2594
## scentFruity      -2.548e-02  9.784e-02  -0.260   0.7946
## scentOriental    -5.623e-02  9.272e-02  -0.606   0.5444
## scentSpicy        5.731e-02  8.634e-02   0.664   0.5070
## scentWoody       -3.487e-02  7.916e-02  -0.440   0.6598
## num_sel_ratings  -7.176e-07  2.515e-06  -0.285   0.7755
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5621 on 777 degrees of freedom
## Multiple R-squared:  0.03631,    Adjusted R-squared:  0.01398
## F-statistic: 1.626 on 18 and 777 DF,  p-value: 0.048
```

```
lm.3 = lm(item_rating ~ old_price, data = perfume)
summary(lm.3)
```

```
##
## Call:
## lm(formula = item_rating ~ old_price, data = perfume)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5465 -0.1295  0.0735  0.4343  0.5152
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.484e+00  3.369e-02 133.071   <2e-16 ***
## old_price   1.379e-04  8.251e-05   1.671   0.0951 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5654 on 794 degrees of freedom
## Multiple R-squared:  0.003505,   Adjusted R-squared:  0.00225
## F-statistic: 2.793 on 1 and 794 DF,  p-value: 0.0951
```

```
lm.4 = lm(old_price ~ item_rating, data = perfume)
summary(lm.4)
```

```
##
## Call:
## lm(formula = old_price ~ item_rating, data = perfume)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -331.72 -184.13  -12.51  131.31 2009.78
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   213.13      69.42   3.070  0.00221 **
## item_rating    25.42      15.21   1.671  0.09510 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 242.8 on 794 degrees of freedom
## Multiple R-squared:  0.003505,   Adjusted R-squared:  0.00225
## F-statistic: 2.793 on 1 and 794 DF,  p-value: 0.0951
```

```
AIC(lm.1)
```

```
## [1] 10919.48
```

```
lm.11 = lm(old_price ~ big_brand + nose_rating + item_rating + ml_order + seller_rating + scent, data =
summary(lm.11)
```

```
##
## Call:
## lm(formula = old_price ~ big_brand + nose_rating + item_rating +
##     ml_order + seller_rating + scent, data = perfume)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -387.70 -161.30  -27.46  122.78 2019.47
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   -103.7250   151.9801  -0.682   0.4951
## big_brand       69.7256    17.4434   3.997 7.01e-05 ***
## nose_rating     -0.7030     0.3185  -2.207   0.0276 *
## item_rating     22.1729    15.0007   1.478   0.1398
## ml_order         8.9983     3.6697   2.452   0.0144 *
## seller_rating   52.1072    34.7491   1.500   0.1341
## scentFloral     62.8510    30.9796   2.029   0.0428 *
## scentFresh     -48.6057    46.3645  -1.048   0.2948
## scentFruity     22.0960    39.9583   0.553   0.5804
## scentOriental    9.1295    37.8535   0.241   0.8095
## scentSpicy     -19.1037    36.1435  -0.529   0.5973
## scentWoody      51.3884    32.9452   1.560   0.1192
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 237.6 on 784 degrees of freedom
## Multiple R-squared:  0.05779,    Adjusted R-squared:  0.04457
## F-statistic: 4.371 on 11 and 784 DF,  p-value: 2.208e-06
```

```
AIC(lm.11)
```

```
## [1] 10981.78
```

```
plot(perfume$item_rating, perfume$old_price)
```