STA6714: Assignment 7 - Online Retail

Sandamini Senaratne

Take the Online Retail data set from UCI Machine Learning Repository.

Online Retail.xlsx Download Online Retail.xlsx

https://archive.ics.uci.edu/ml/datasets/online+retailLinks to an external site.

The goal of your project:

Your goal is to get the customer to add an additional item to their invoice. You need to create a model that will consider which items are currently being ordered, then recommend an item to be added to their shopping cart.

For this assignment, you will submit data sets, the recommended items, and a summary table:

Recommendation data set:

The recommendation data set will have two columns:

the invoice number (id number) in the first column, and the description of the recommended item in the second column. Each invoice needs to appear once in your data set. The description of the recommended item needs to be as it appears in the original data. The recommended item must not be on the invoice.

Summary table:

The summary table will be a table with three columns:

The description of the recommended items as it appears in the original data (first column) The count of the times the item appears in an invoice (2nd column) The count of the number of times your model recommended the item (3rd column) Things to consider:

Your data has a lot of distinct items for purchase. To build an effective model, you need to group similar items together before modeling, but you need to recommend only one item per invoice.

Your data is in long format by item and invoice. You will need to reshape your data before modeling. You will want wide format with each invoice corresponding to one row.

Your data has over half a million rows. Good data prep is needed to make this assignment computationally feasible.

Loading the libraries

```
library(readxl)
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)
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ forcats   1.0.0     ✓ readr     2.1.4
## ✓ ggplot2   3.4.1     ✓ stringr   1.5.0
## ✓ lubridate 1.9.2     ✓ tibble    3.2.1
## ✓ purrr     1.0.1
```

```
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force
all conflicts to become errors
```

```
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(stringdist)
```

```
##
## Attaching package: 'stringdist'
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(writexl)
```

Loading the dataset

```
setwd("C:/Users/sanda/OneDrive/Documents/DataPrep")
df <- readxl::read_excel("Online Retail.xlsx")

#dropping the rows with na values
df <- df[!is.na(df$Description),]
#removing duplicates
df <- distinct(df)
#dropping rows with negative quantity
df <- df[df$Quantity>=0,]

df1 <- df #keeping for lateruse
```

String manipulation

```
#Convert all letters to lowercase.
df$Description <- stringr::str_to_lower(df$Description)

#remove punctuations and numbers
df$Description <- stringr::str_replace_all(
  string = df$Description,
  pattern = "[:punct:]",
  replacement = " "
)
df$Description <- stringr::str_replace_all(
  string = df$Description,
  pattern = "[:digit:]",
  replacement = " "
)
df$Description <- stringr::str_replace_all(
  string = df$Description,
  pattern = "\\W",
  replacement = " "
)

#Get rid of extra white space.
df$Description <- trimws(stringr::str_replace_all(
  string = df$Description,
  pattern = "\\s+",
  replacement = " "
))

#Remove stop words
df$Description <- tm::removeWords(
  x = df$Description,
  words = tm::stopwords(kind = "SMART")
)
```

Finding the most frequent item list

```
#getting a subset of the wanted variables
FreqItem <- subset(df, select = c("StockCode","Description","Quantity"))
#getting the total quantities by grouping
FreqItem <- FreqItem %>%
  group_by(StockCode, Description) %>%
  summarise(Quantity = sum(Quantity))

## `summarise()` has grouped output by 'StockCode'. You can override using
the
## `.groups` argument.

FreqItem <- arrange(FreqItem, desc(Quantity))

# getting the list of items with their cumulative quantities
v_sum <- sum(FreqItem$Quantity)
FreqItem$CumuSum <- cumsum(FreqItem$Quantity)
v_names <- FreqItem$Description
#the items that make up 80% of the purchases
FreqItem80 <- FreqItem[((FreqItem$CumuSum)/v_sum) <= 0.8, ]
FreqItem80$CumuSum <- NULL
v_names80 <- FreqItem80$Description
```

String distances

```
stringItem <- sapply(
  X = v_names80,
  FUN = function(x){
    stringdist::stringdist(
      a = x,
      b = FreqItem80$Description,
      method = "qgram"
    )
  }
)
#converting to a matrix to get the clusters under hclustering
M <- as.matrix(stringItem)
```

Hierarchical clustering

```
hclust_StringItem <- hclust(as.dist(stringItem))
cutree_M <- cutree(
  tree = hclust_StringItem,
  k = 100
)
cutree_df <- as.data.frame(cutree_M)
```

Getting the dataset with most frequent items

```
#binding the cluster column to FreqItem80 dataset
FreqItem80 <- cbind(FreqItem80, cutree_df)
```

```r
colnames(FreqItem80)[4] <- "Cluster"

clustData <- subset(FreqItem80, select = c("StockCode", "Cluster"))
# organizing items by group and quuantity
FreqItem80 <- arrange(FreqItem80, Cluster, desc(Quantity))

#getting the most freq data items in the original dataset
join_df <- inner_join(
  x = df1,
  y = clustData,
  by = "StockCode"
)
```

```
## Warning in inner_join(x = df1, y = clustData, by = "StockCode"): Detected
an unexpected many-to-many relationship between `x` and `y`.
## ℹ Row 1 of `x` matches multiple rows in `y`.
## ℹ Row 307 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

Recommendation dataset

```r
#grouping data by invoice
invoice_data <- join_df %>%
  group_by(InvoiceNo)

#creating empty dataset for recommendations
recommendations <- data.frame(
  InvoiceNo = character(),
  RecommendedItem = character(),
  stringsAsFactors = FALSE
)

#looping over each invoice
for (i in unique(join_df$InvoiceNo)) {

  #getting items purchased in invoice
  items_purchased <- join_df %>%
    filter(InvoiceNo == i) %>%
    select(Description) %>%
    distinct()

  #getting list of recommended items
  recommended_items <- setdiff(v_names80, items_purchased$Description)

  #selecting one recommended item
  if (length(recommended_items) > 0) {
    recommended_item <- sample(recommended_items, 1)
```

```
    #adding recommendation to data frame
    recommendations <- rbind(recommendations, data.frame(
      InvoiceNo = i,
      RecommendedItem = recommended_item,
      stringsAsFactors = FALSE
    ))
  }
}

head(recommendations)
```

```
##   InvoiceNo                RecommendedItem
## 1    536365 single antique rose hook ivory
## 2    536366   set red retrospot tea towels
## 3    536367      pink doughnut trinket pot
## 4    536368          hand   chocolate sign
## 5    536370  bundle  school exercise books
## 6    536371           camouflage led torch
```

```
# Write the recommendations dataset to a file
write.csv(recommendations, "recommendationsdataset.csv", row.names = FALSE)
```

Summary table

```
#counting the number of times each item appears in an invoice
item_count <- df %>%
  group_by(Description) %>%
  summarise(count = n())

#counting the number of times each item was recommended
recommendation_count <- recommendations %>%
  group_by(RecommendedItem) %>%
  summarise(count = n())

colnames(recommendation_count)[1] <- "Description"

#combining the item count and recommendation count into a summary table
summary_table <- item_count %>%
  left_join(recommendation_count, by = "Description") %>%
  replace(is.na(.), 0) %>%
  rename("Item" = "Description", "Invoice Count" = "count.x", "Recommendation
Count" = "count.y") %>%
  arrange(desc(`Recommendation Count`))

head(summary_table)
```

```
## # A tibble: 6 × 3
##   Item                               `Invoice Count` `Recommendation
```

```
Count`
##   <chr>                                      <int>
<int>
## 1 pink polkadot cup                            310
39
## 2 wrap alphabet design                         235
36
## 3 pack  london tissues                         524
35
## 4 alphabet stencil craft                       186
34
## 5 ceramic strawberry cake money bank           395
34
## 6 doorstop retrospot heart                     296
34

# Write the summary table to a file
write.csv(summary_table, "summarytable.csv", row.names = FALSE)
```