Assignment 03 Template

Darren Luwi (29051754)

20/09/2021

This is the initial setup for your assignment. Please do not change the following three chunks.

```
library(tidyverse)
library(ggplot2)
library(here)
library(ggdendro)
library(ade4)
library(arules)
library(factoextra)
```

```
set.seed(372910)
```

PART A

In this section, we are interested in two cluster sets: US and Non-US films on Netflix.

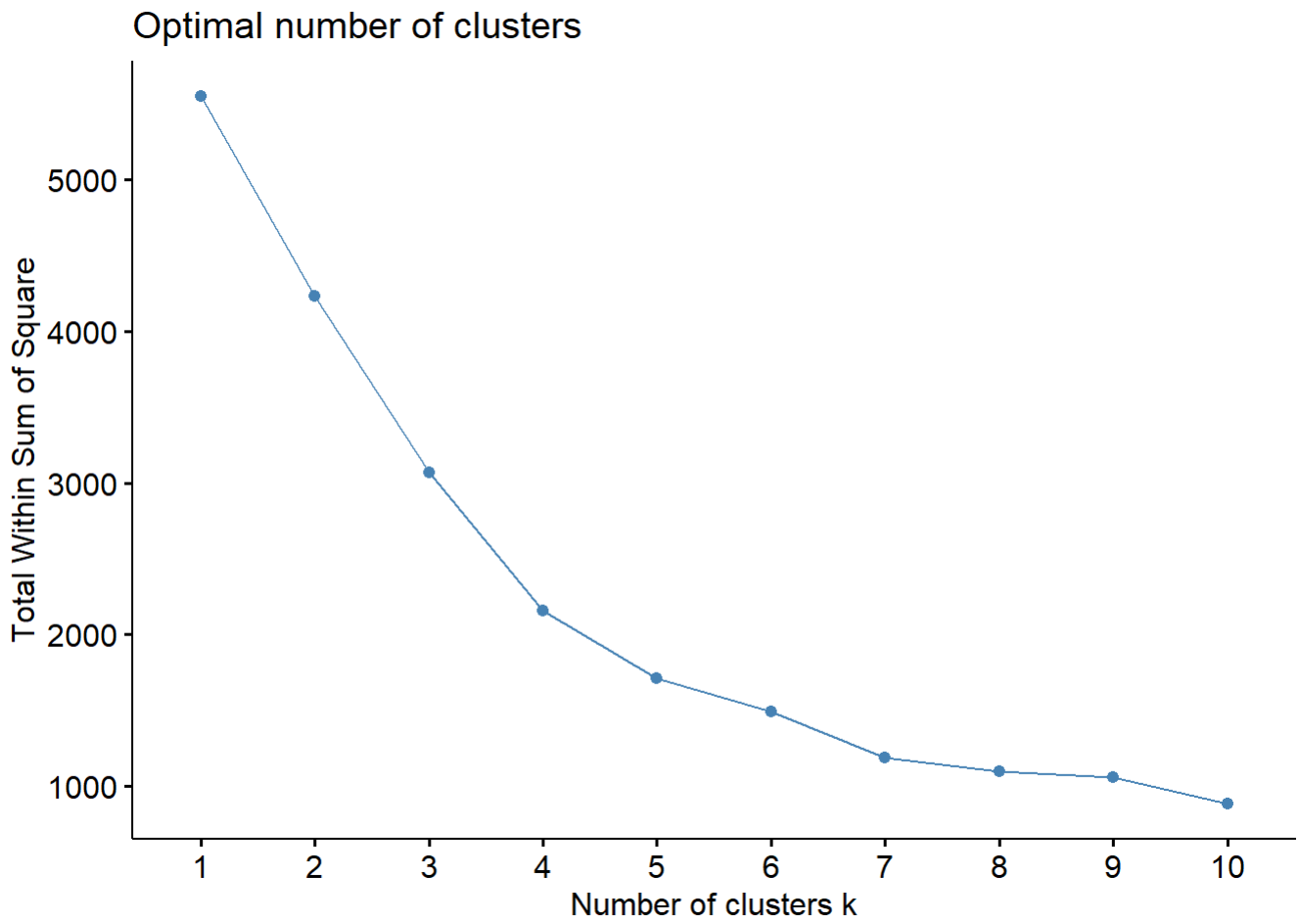
As we all know, the United States are capable of producing top-notch and high on-demand films, with many considering the country as the best producer of films. However, recently the market for Non-US films began growing, with the number of audience interested in these films as well as the hype for it rapidly rising. In addition, the fact that some non-US films, most notably “Parasite” and “Minari”, has claimed accolades at Oscars and other awarding ceremonies should act as a concrete evidence that foreign countries are also competent in terms of producing quality films. Therefore, this analysis aims to clarify whether Non-US and US films are indeed similar, or are different to some extent, or even completely.

Initially, two objects, netflix_usfilms and netflix_nonusfilms, were created to represent US and Non-US Netflix films respectively, with both containing variables ‘min_age_rating’ (Minimum Age Rating), ‘years_old’ (Age of film), and ‘length’ (Length of film). These are the three most commonly discussed attributes of a film (the others being film rating and revenues, which, unfortunately, were not provided on the raw dataset), thus justifying their inclusion in this analysis. Meanwhile, it was assumed that all NA values observed for Minimum Age Rating simply means that the corresponding film does not have any age rating, thus all NAs have been replaced with 0.

```
netflix_df <- read_csv(here("data/netflix_metadata.csv"))
netflix_df[is.na(netflix_df)] <- 0
view(netflix_df)

netflix_usfilms <- netflix_df %>% filter((movie == '1') & (united_states == '1')) %>% drop_na()
netflix_nonusfilms <- netflix_df %>% filter((movie == '1') & (united_states == '0')) %>% drop_na
()

netflix_usfilms <- netflix_usfilms %>%
  select(min_age_rating, years_old, length) %>%
  scale()

netflix_nonusfilms <- netflix_nonusfilms %>%
  select(min_age_rating, years_old, length ) %>%
  scale()

view(netflix_usfilms)
view(netflix_nonusfilms)
```

Next, it was decided that k-means clustering will be utilized, considering this clustering method works better with extremely large datasets. Before conducting k-means clustering, two separate Elbow Plots (one under this section, the other on Appendix 1) were plotted to help us determine the optimal number of clusters (k) to use for both cluster sets. Here, although the choice of number of clusters is subjective, but as both Elbow Plots visualized that at k=4 the plots begin bending inwards, we assumed that 4 is the ideal number of clusters.

```
set.seed(372910)
fviz_nbclust(netflix_usfilms, kmeans,method = "wss")
```




```
km_usfilms <- kmeans(netflix_usfilms, centers = 4, nstart = 25)
km_nonusfilms <- kmeans(netflix_nonusfilms, centers = 4, nstart = 25)
```

Finally, two scatterplots (one for netflix_usfilms & another for netflix_nonusfilms) were plotted under Appendix 2 and 3, respectively. This is where profiles for each clusters on each cluster sets were built:

netflix_usfilms clusters: (1) = Longer duration, Higher age rating, older (2) = Longer duration, Higher age rating, less old (3) = Shorter duration, Higher age rating, less old (4) = Shorter duration, Lower age rating,less old

netflix_nonusfilms clusters: (1) = Shorter duration, Higher age rating, less old (2) = Shorter duration, Lower age rating, less old (3) = Longer duration, Higher age rating, older (4) = Longer duration, Higher age rating, older

Interestingly,a somewhat similar pattern can be seen from the two scatterplots: one cluster dominating the left-most side of the first grid (i.e. the lowest age rating), a few observations seen on the second and third grid, and three clusters occupying the remaining grids, where here each cluster taking fairly equal shares of spaces on each grid, with one cluster nucleated on the upper half (indicating longer duration), another centered on the lower half (indicating shorter duration), and the final one being more spread out. Another appealing feature is that there are way more points under the Non-US films plot, indicating that there are more Non-US than US films on Netflix. Finally, some outliers can also be seen.

Next, the summary table for the US films cluster set suggests that cluster 2 has the highest mean length, with most films here having an average runtime of 117.25 minutes. Meanwhile, cluster 1 has the highest mean age, being the ‘oldest’ cluster with an average film age of 44.38. Finally, cluster 3 has the highest minimum age rating, with an average of 16.1.

```
netflix_usfilms1 <- netflix_df %>%
  filter((united_states == '1') & (movie == '1')) %>% drop_na()
netflix_usfilms1 <- netflix_usfilms1 %>%
  select(length,years_old, min_age_rating)
netflix_usfilms1 %>%
  mutate(kmeans = km_usfilms$cluster) %>%
  group_by(kmeans) %>%
  summarise(meanLength = mean(length), meanAge = mean(years_old), meanMinAgeRating = mean(min_ag
e_rating))
```

```
## # A tibble: 4 x 4
##   kmeans meanLength meanAge meanMinAgeRating
##    <int>      <dbl>   <dbl>            <dbl>
## 1      1       93.4   44.4              8.78
## 2      2      117.    11.0             15.5
## 3      3       82.9    4.53            16.1
## 4      4       76.9    6.81             0.862
```

On the other hand, for the Non-US films cluster set, cluster 3 has the highest mean length with 132.93, slightly surpassing cluster 4. Cluster 4 has the highest mean Age with 37.66, and cluster 1 has the highest mean minimum age rating with 15.95.

```
netflix_nonusfilms1 <- netflix_df %>%
  filter((united_states == '0') & (movie == '1')) %>% drop_na()
netflix_nonusfilms1 <- netflix_nonusfilms1 %>%
  select(length,years_old, min_age_rating)
netflix_nonusfilms1 %>%
  mutate(kmeans = km_nonusfilms$cluster) %>%
  group_by(kmeans) %>%
  summarise(meanLength = mean(length), meanAge = mean(years_old), meanMinAgeRating = mean(min_ag
e_rating))
```

```
## # A tibble: 4 x 4
##   kmeans meanLength meanAge meanMinAgeRating
##    <int>      <dbl>   <dbl>            <dbl>
## 1      1       93.2    5.15            16.0
## 2      2       87.8    5.88             0.811
## 3      3      133.     7.74            15.0
## 4      4      128.    37.7             11.4
```

The two summary tables for both cluster sets suggests that, while the Non-US films have a longer average runtime,US films are slightly older, and both cluster sets have slightly similar average minimum age rating.

```
netflix_usfilms1 %>%
  select(length, years_old, min_age_rating) %>%
  summarise(meanLength = mean(length), meanAge = mean(years_old), meanMinAgeRating = mean(min_ag
e_rating))
```

```
## # A tibble: 1 x 3
##   meanLength meanAge meanMinAgeRating
##        <dbl>   <dbl>            <dbl>
## 1       89.4    8.68             12.0
```

```
netflix_nonusfilms1 %>%
  select(length, years_old, min_age_rating) %>%
  summarise(meanLength = mean(length), meanAge = mean(years_old), meanMinAgeRating = mean(min_ag
e_rating))
```

```
## # A tibble: 1 x 3
##   meanLength meanAge meanMinAgeRating
##        <dbl>   <dbl>            <dbl>
## 1       105.    7.77             12.5
```

Moving on, we wish to test how stable the clustering method are for the two cluster sets. We do this by randomly selecting 500 US and Non-US films, before repeating k-means clustering on each, and compare the scatterplots generated (Appendix 4 & 5), against the initial US films and Non-US films scatterplots (Appendix 2 & 3).

```
set.seed(372910)
netflix_usfilms1 <- netflix_usfilms1[sample(nrow(netflix_usfilms1),500),] %>% scale
view(netflix_usfilms1)
km_usfilms1 <- kmeans(netflix_usfilms1, centers = 4, nstart = 25)

netflix_nonusfilms1 <- netflix_nonusfilms1[sample(nrow(netflix_nonusfilms1),500),] %>% scale
view(netflix_nonusfilms1)
km_nonusfilms1 <- kmeans(netflix_nonusfilms1, centers = 4, nstart = 25)
```

Based on the plots, new profiles can be created:

netflix_usfilms1 clusters: (1) = Longer duration, Lower age rating, Older (2) = Shorter duration, Higher age rating, Less old (3) = Shorter duration, Lower age rating, Less old (4) = Longer duration, Higher age rating, Less old

netflix_nonusfilms1 clusters: (1) = Longer duration, Higher age rating, Older (2) = Longer duration, Higher age rating, Less old (3) = Shorter duration, Higher age rating, Less old (4) = Shorter duration, Lower age rating, Less old

Apart from the cluster numbers assigned, we can see that there is only one dissimilarity between the profiles of the train set (netflix_usfilms1 & netflix_nonusfilms1) and the test set (netflix_usfilms & netflix_nonusfilms). Simultaneously, similar plot distributions can be observed. Overall, it can be concluded that the clusters are stable.

In relation to this, we can assume that k-means clustering is the better method. Assuming we are producing similar analysis with similar datasets in the future, k-means would perform better, as it works well with large datasets, can easily adapt to new examples, is relatively simple to implement, as well as having a better algorithm to produce clusters. The only downside of k-means clustering is that the number of cluster k must be manually selected.

In conclusion, as both cluster sets have somewhat similar cluster profiles and characteristics, it can be confirmed that the US and Non-US films are similar. Netflix has done an extremely great job by featuring a lot of films, and this analysis should encourage Netflix to improve more by featuring not only more US films, but also foreign films, as it is safe to assume that viewers are not really concerned on the origin of films, but on their quality. However, the fact that both cluster sets are relatively “young” (i.e. having relatively low ages) and have relatively low mean

minimum age rating does not necessarily mean Netflix must focus on newer and low minimum age rating films only, but also on older and various minimum age rating films. Doing these will not only bring monetary benefits, but also promote diversity, and possibly attract more foreign production studios or streaming services to partner up with the company.

PART B

This section concerns the production of film recommendations for Netflix users. This can be considered as one of the most important feature on Netflix, at it aims to improve user experience by easing the process of finding suitable films.

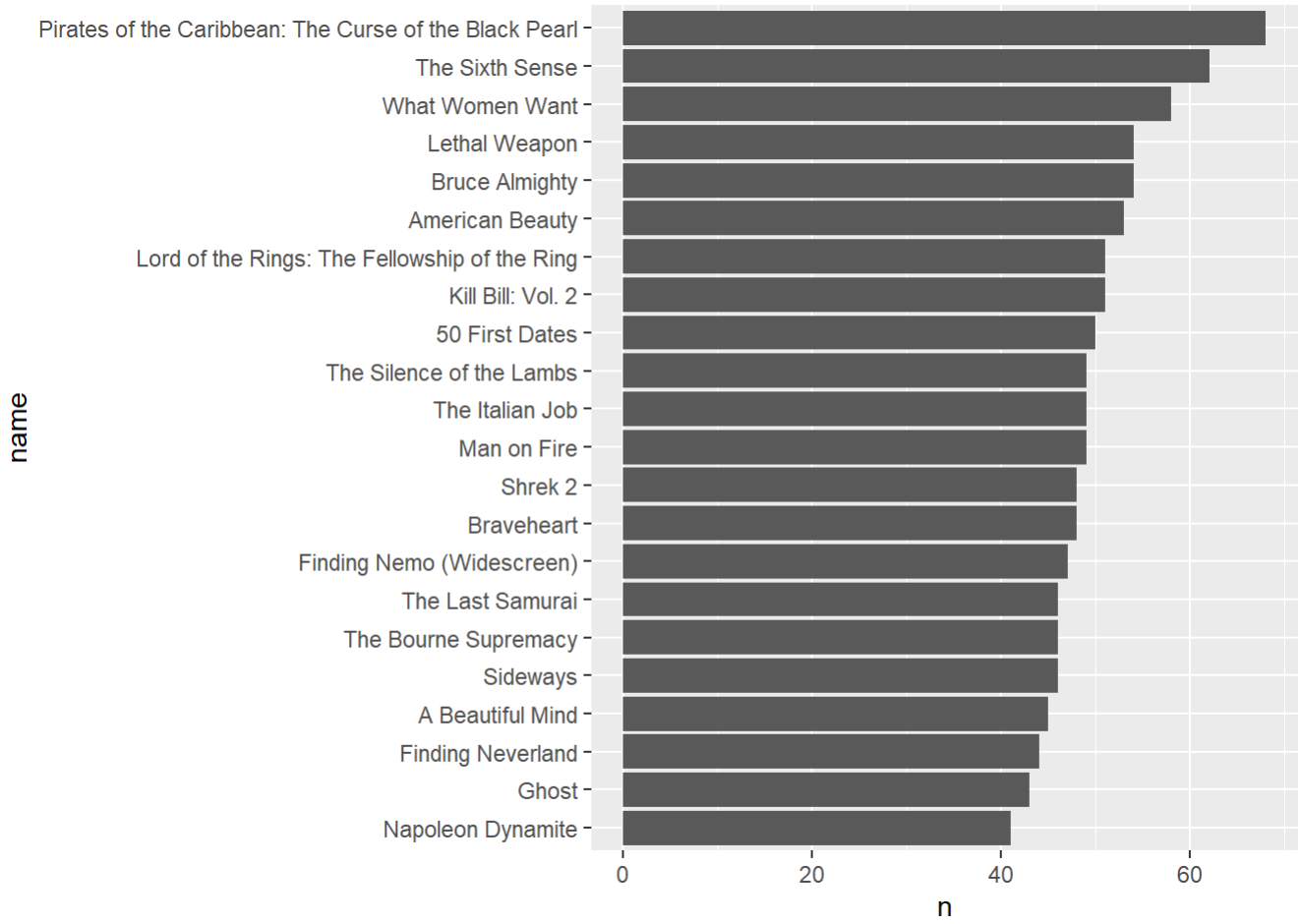
The variable ‘user_id’ has been converted into a factor to prevent R from classifying it as numeric.

```
user_df <- read.csv(here("data/user_data.csv"))
view(user_df)
user_df <- user_df %>%
  mutate(user_id = factor(user_id))
```

Before proceeding, bear in mind that it is very likely for some films to be more frequent than other films. This means that we should be careful when generating and interpreting plausible rules.

The bar plot below suggests that “Pirates of the Carribbean: The Curse of the Black Pearl” is the most common film, with a frequency greater than 60.

```
user_df %>%
  group_by(name) %>%
  summarise(n = n()) %>%
  ungroup() %>%
  filter(n>mean(n))%>%
  mutate(name = reorder(name, n)) %>%
  ggplot(aes(x = name, y = n)) +
  geom_col() + coord_flip()
```



We then proceed by creating a list of movies that each users have viewed. This was done via the 'purrr' package, converting the dataframe into a transaction object, and the 'eclat' function with a support of 0.3. Although it is preferable to have higher support (higher support = rules generated will be more applicable in large number of future transactions), 0.3 is the highest we can go here, as figures greater than that will be futile.

```
playlist <- user_df %>%
  group_by(user_id) %>%
  group_split()
playlist <- playlist %>%
  purrr::map(function(x) x %>%
               select(name) %>%
               unique() %>%
               deframe())
playlist_transactions <- as(playlist, "transactions")
frequentItems <- eclat(playlist_transactions, parameter = list(supp = 0.3))
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE     0.3      1     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 45
##
## create itemset ...
## set transactions ...[52 item(s), 150 transaction(s)] done [0.00s].
## sorting and recoding items ... [19 item(s)] done [0.00s].
## creating bit matrix ... [19 row(s), 150 column(s)] done [0.00s].
## writing  ... [19 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

Rule mining was then conducted with the 'apriori' function. A minimum support of 0.2, which, again is the highest support we can go, was used, and a relatively high minimum confidence of 0.8 was utilized to ensure that the algorithm is highly 'confident' with the accuracy of the rules generated.

```
playlist_rules <- apriori(playlist_transactions, parameter = list(supp = 0.2, conf = 0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.2      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 30
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[52 item(s), 150 transaction(s)] done [0.00s].
## sorting and recoding items ... [46 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [19 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

As a result, 19 rules were generated, which is reasonable, considering the level of support and confidence used.

We then place the rules into a dataframe.

```
playlist_rules <- DATAFRAME(playlist_rules)
```

And finally, we filter the dataframe to only consider rules where the lift is greater than 2. In market basket analysis, higher lifts are often preferred, as it translates to stronger link between the antecedent and the consequent, preventing generation of ineffective rules due to extremely common observations (for instance “Pirates of the Carribean: The Curse of the Black Pearl”). However, a lift of 2 is the greatest we can go here, as a lift of 3 onwards generates 0 rules.

```
playlist_rules %>%
  filter(lift > 2)
```

```
##                                                                                LHS
## 1                                                                {Being John Malkovich}
## 2                                           {A Beautiful Mind,Finding Nemo (Widescreen)}
## 3 {Finding Nemo (Widescreen),Pirates of the Caribbean: The Curse of the Black Pearl}
## 4          {A Beautiful Mind,Pirates of the Caribbean: The Curse of the Black Pearl}
## 5                {Braveheart,Pirates of the Caribbean: The Curse of the Black Pearl}
## 6  {Pirates of the Caribbean: The Curse of the Black Pearl,The Silence of the Lambs}
## 7                     {Bruce Almighty,Lord of the Rings: The Fellowship of the Ring}
##                                                          RHS   support confidence
## 1                                          {American Beauty} 0.2000000  0.8333333
## 2 {Pirates of the Caribbean: The Curse of the Black Pearl} 0.2066667  0.9393939
## 3                                          {A Beautiful Mind} 0.2066667  0.8611111
## 4                               {Finding Nemo (Widescreen)} 0.2066667  0.8157895
## 5                                           {The Sixth Sense} 0.2133333  0.8648649
## 6                                           {The Sixth Sense} 0.2200000  0.8918919
## 7 {Pirates of the Caribbean: The Curse of the Black Pearl} 0.2000000  0.9090909
##    coverage     lift count
## 1 0.2400000 2.358491    30
## 2 0.2200000 2.072193    31
## 3 0.2400000 2.870370    31
## 4 0.2533333 2.603583    31
## 5 0.2466667 2.092415    32
## 6 0.2466667 2.157803    33
## 7 0.2200000 2.005348    30
```

As a result, 7 rules were generated.

Theoretically, there are a lot of factors which can explain these rules, such as similarities in genres of the antecedent and consequent, similarities in runtime, actors featured, director, production studio, etc. However, based on the table above, a recommendation rule can be generated: Based on similarities in film genre and release date.

Film genre is one of the influential factor that links the antecedents (i.e. list on the left hand side) and consequent (i.e. list on the right hand side) here. For instance, users who watched “Being John Malkovich” will watch “American Beauty” due to both being comedy films, and similarly, those who watched “Bruce Almighty” and “Lord of the Rings: The Fellowship of the Ring” will watch “Pirates of the Carribean: The Curse of the Black Pearl” since they are all fantasy films.

The second rule-influencing factor is the release date. For example, “Being John Malkovich” and “American Beauty” were released in the 90s, thus being linked. Similarly, “Finding Nemo”, “Pirates of the Carribean: The Curse of the Black Pearl” and “A Beautiful Mind” were are linked as they were released in the early 2000s.

The fact that all rules have supports lying around 0.2 suggests that approximately 20% of transactions contains all the films in the rule itemsets. This is reasonably large, considering the size of the dataset.

Next, the confidence level of all the rules are also high, suggesting that given a user watches the films listed on the antecedent, there is a high chance that he/she will also watch the film listed on the consequent.

Finally, all rules have lifts around 2, implying a relatively great link between the antecedent films and the consequent film.

Next, we wish to test whether the recommendation rule is in fact effective. This was done by computing the number of low ratings users gave to each consequent films, before adding all of them into object ‘filmratings’.

As a threshold, we consider ratings below 3 as low.

```
rule1antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name == "Being John Malkovich")
view(rule1antecedent)

rule1consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "American Beauty")
view(rule1consequent)

rule1 <- right_join(rule1antecedent, rule1consequent)
view(rule1)


rule2antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("A Beautiful Mind", "Finding Nemo (Widescreen)"))
view(rule2antecedent)

rule2consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "Pirates of the Caribbean: The Curse of the Black Pearl")
view(rule2consequent)

rule2 <- right_join(rule2antecedent, rule2consequent)
view(rule2)


rule3antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("Finding Nemo (Widescreen)", "Pirates of the Caribbean: The Curse of the Bl
ack Pearl"))
view(rule3antecedent)

rule3consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "A Beautiful Mind")
view(rule3consequent)

rule3 <- right_join(rule3antecedent, rule3consequent)
view(rule3)


rule4antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("A Beautiful Mind", "Pirates of the Caribbean: The Curse of the Black Pear
l"))
view(rule4antecedent)

rule4consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "Finding Nemo (Widescreen)")
view(rule4consequent)
```

```
rule4 <- right_join(rule4antecedent, rule4consequent)
view(rule4)


rule5antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("Braveheart", "Pirates of the Caribbean: The Curse of the Black Pearl"))
view(rule5antecedent)

rule5consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "The Sixth Sense")
view(rule5consequent)

rule5 <- right_join(rule5antecedent, rule5consequent)
view(rule5)


rule6antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("Pirates of the Caribbean: The Curse of the Black Pearl", "The Silence of t
he Lambs"))
view(rule6antecedent)

rule6consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "The Sixth Sense")
view(rule6consequent)

rule6 <- right_join(rule6antecedent, rule6consequent)
view(rule6)


rule7antecedent <- user_df %>%
  select(user_id, name) %>%
  filter(name %in% c("Bruce Almighty", "Lord of the Rings: The Fellowship of the Ring"))
view(rule7antecedent)

rule7consequent <- user_df %>%
  select(user_id, name, rating) %>%
  filter(name == "Pirates of the Caribbean: The Curse of the Black Pearl")
view(rule7consequent)

rule7 <- right_join(rule7antecedent, rule7consequent)
view(rule7)

filmratings <- rule1$rating
filmratings <- rule2$rating
filmratings <- rule3$rating
filmratings <- rule4$rating
filmratings <- rule5$rating
filmratings <- rule6$rating
```

```
filmratings <- rule7$rating
view(filmratings)
```

The following function shows that only 2 users disliked the consequent.

```
filmratings[filmratings<3]
```

```
## [1] 2 2
```

In fact, the average rating for the consequents is 4.22, which is quite high.

```
mean(filmratings)
```

```
## [1] 4.220588
```

Thus, we conclude that the rules generated are in fact useful. Netflix is strongly advised to utilize the recommendation rule, or, in other words, recommend films to users based on the genres and release dates of the films they have watched.

Similarly, Netflix can further enhance its recommendation algorithm by generating rules based on combination of variety of factors, mainly the ones not covered on the dataset, for example genres + film director, or release date + runtime + actors featured. Obviously there are risks associated with this, mainly generation of low-rated consequent, which ultimately affects user experience, but if the expansions is a success, it will definitely make users' lives way easier.

PART C

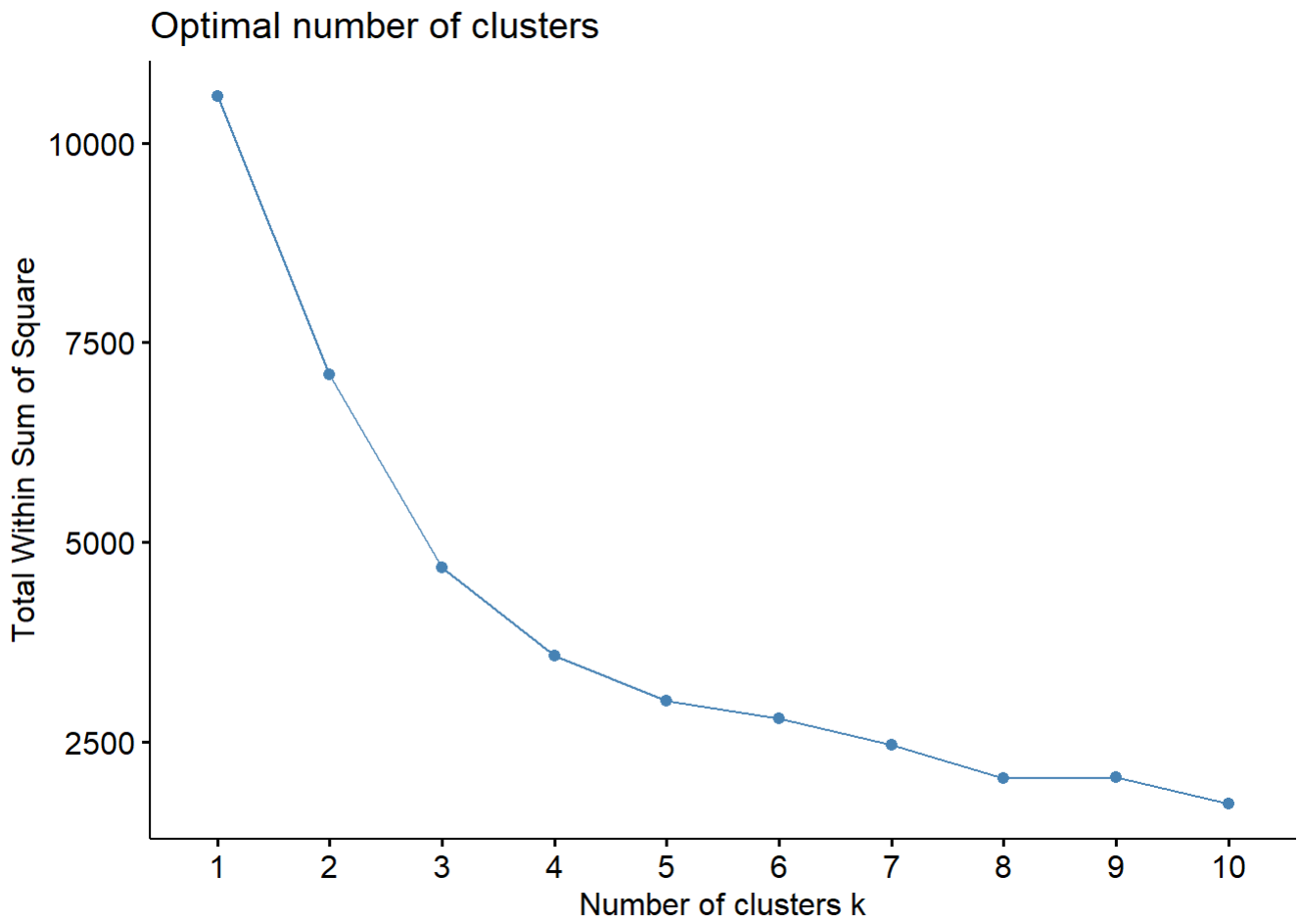
As Netflix is a film streaming service, its main objectives, apart from generating profits, is to provide quality and satisfactory services to its users. Although the clustering analysis performed on the movies on Part A is fancy and useful to some extent, at the end of the day, as its name suggests, it only generates clusters, which is only effective to study the characteristics of the films. Thus, simply based on the given context, Netflix should focus on Market Basket Analysis instead, since it provides an insight on customer behavior, assisting Netflix to improve its services better. If, however, the clustering analysis on Part A was performed on customers instead of movies, it will in fact be as useful, or even more useful, than Part B's market basket analysis.

However, assuming that Netflix insists on using both styles of analysis, Netflix should generate clusters of customer groups instead of films, as, again, it is useful for customer behavior studies.

Finally, assuming that Netflix wishes to invest on additional data collection, data relevant to films, for instance film director and actor name, as well as user-related data like their country and age should be added. All of these can be utilized to understand the link between customer demographics and the films they watch, and simultaneosly generate better film recommendation algorithms.

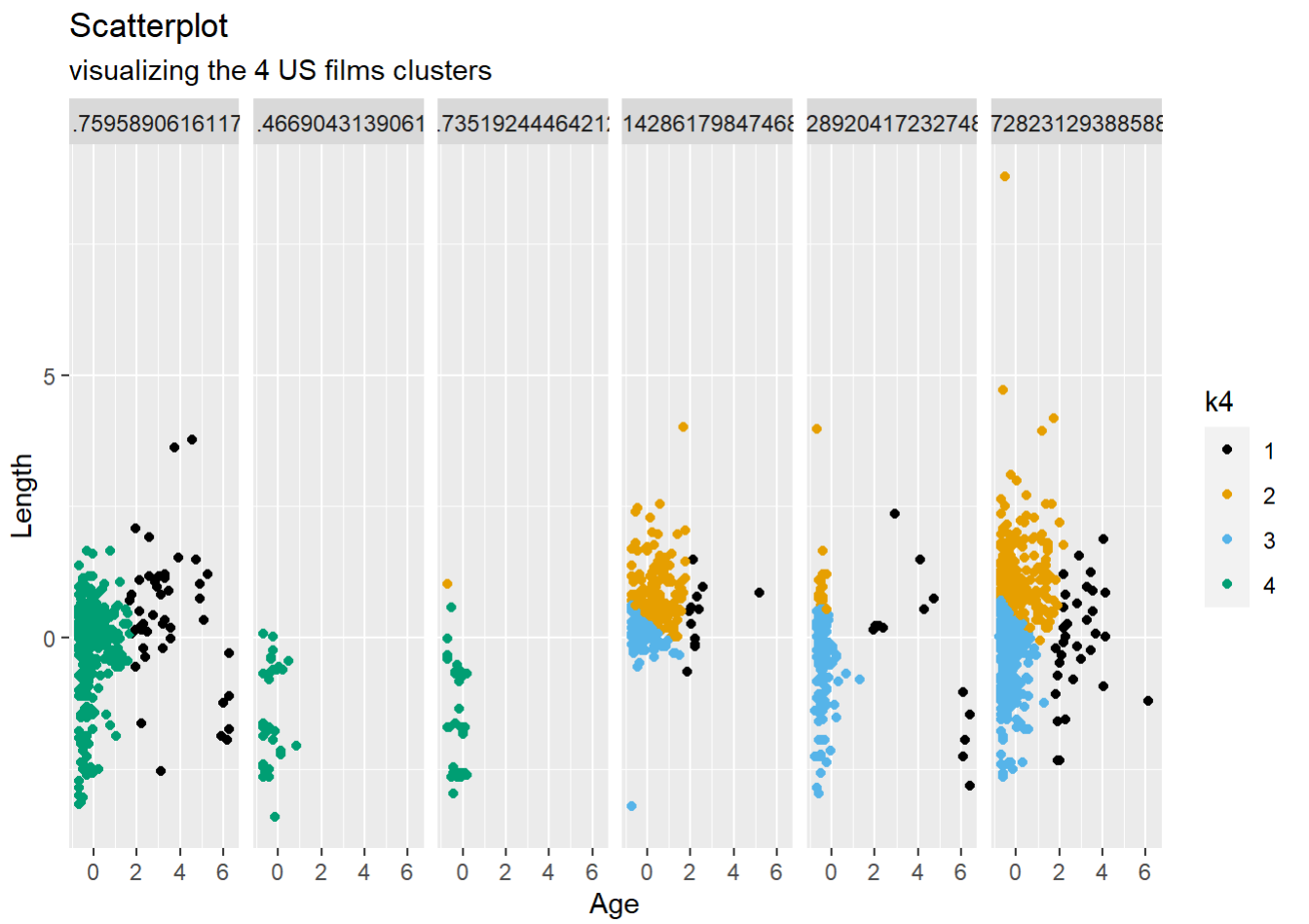
Appendices: Appendix 1. Elbow Plot of netflix_nonusfilms

```
set.seed(372910)
fviz_nbclust(netflix_nonusfilms, kmeans,method = "wss")
```



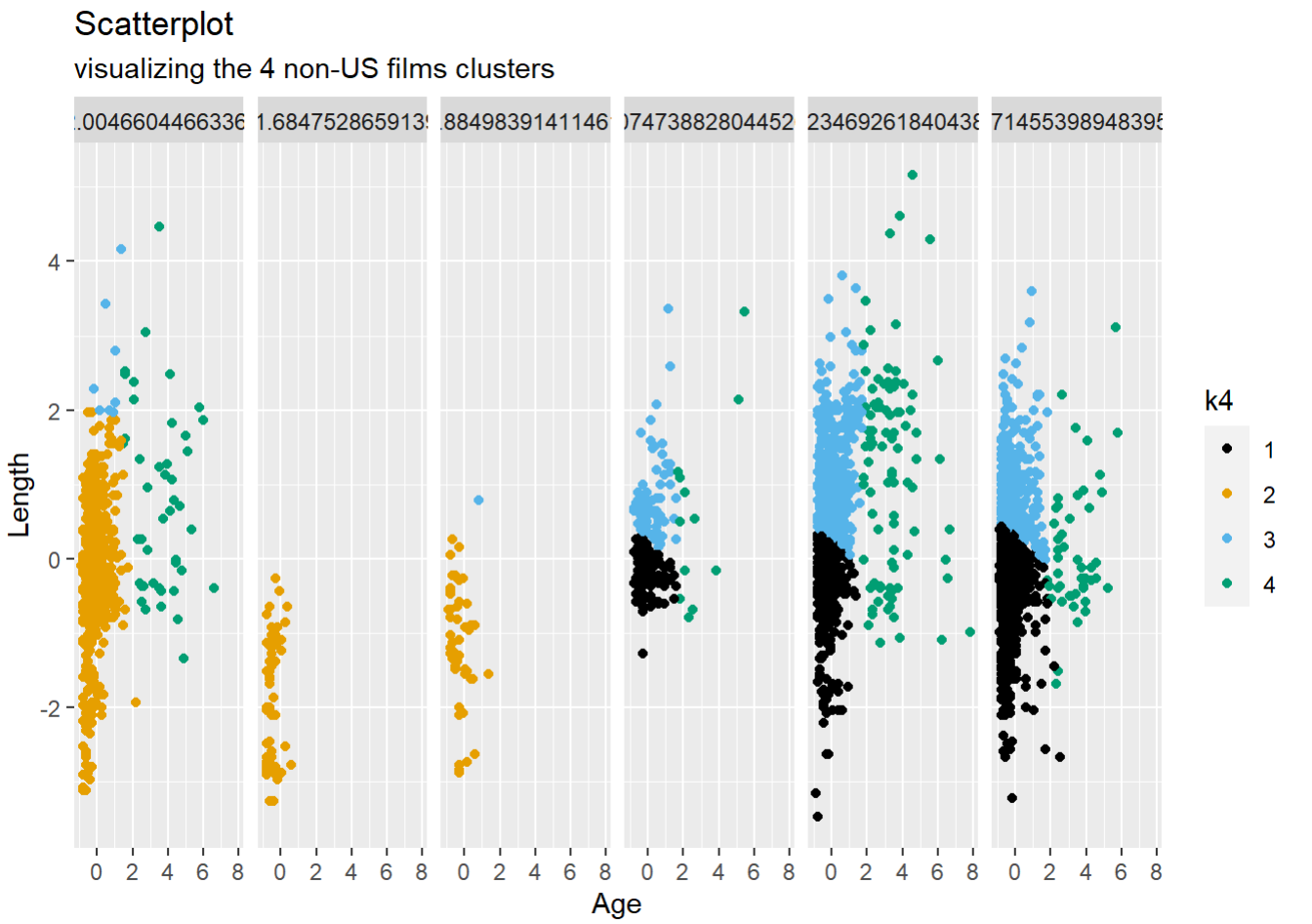
Appendix 2. Scatterplot of netflix_usfilms

```
netflix_usfilms %>%
  as.data.frame() %>%
  mutate(k4 = factor(km_usfilms$cluster)) %>%
  ggplot(data = ., aes(x = years_old , y = length, colour = k4)) + geom_point() + ggthemes::scal
e_color_colorblind() + facet_grid(~min_age_rating)+ labs(title = "Scatterplot", subtitle = "visu
alizing the 4 US films clusters", x = "Age", y= "Length")
```



Appendix 3. Scatterplot of netflix_nonusfilms

```
netflix_nonusfilms %>%
  as.data.frame() %>%
  mutate(k4 = factor(km_nonusfilms$cluster)) %>%
  ggplot(data = ., aes(x = years_old , y = length, colour = k4)) + geom_point() + ggthemes::scal
e_color_colorblind() + facet_grid(~min_age_rating) + labs(title = "Scatterplot", subtitle = "vis
ualizing the 4 non-US films clusters", x = "Age", y= "Length")
```

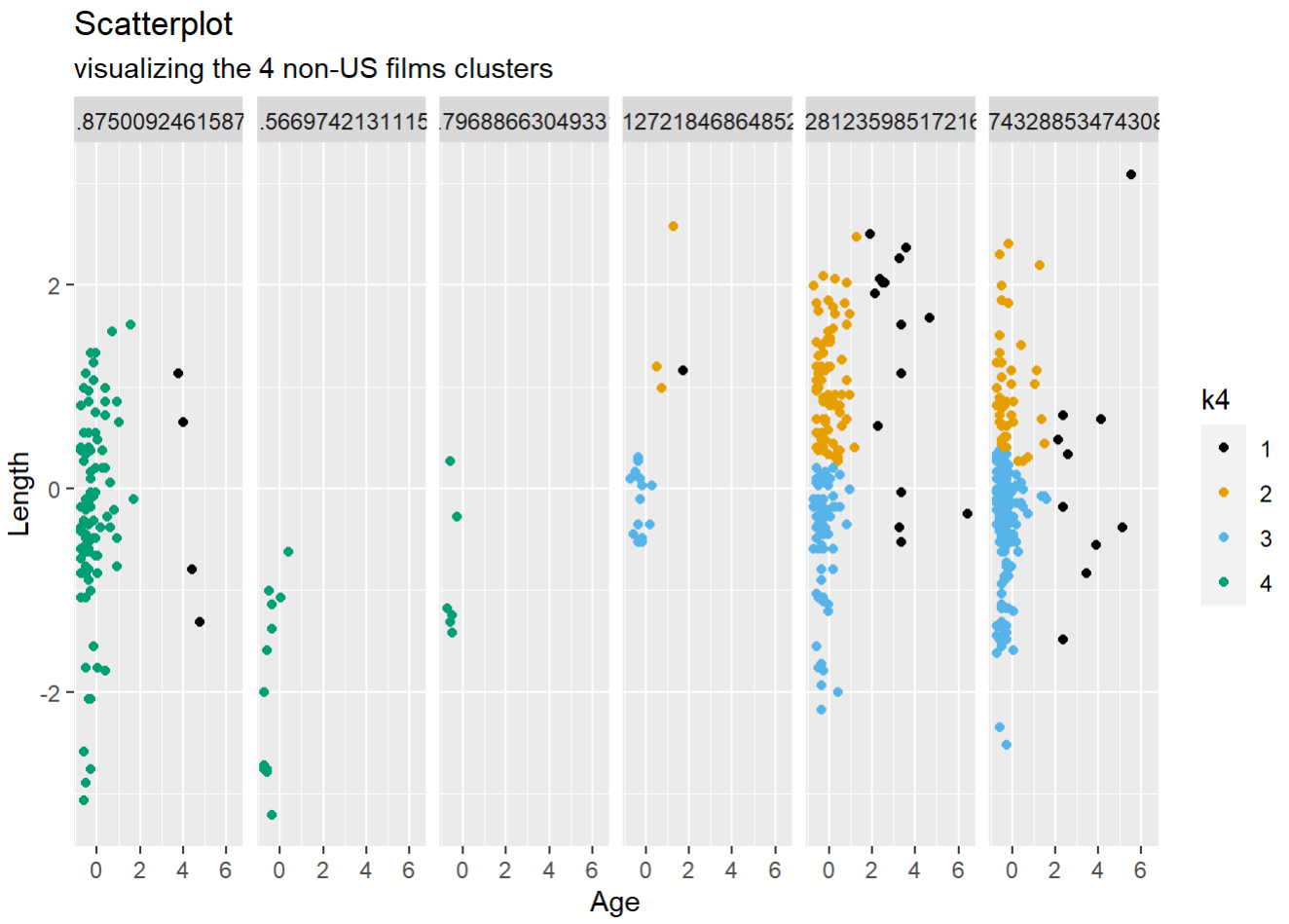


Appendix 4. Scatterplot of netflix_usfilms1

```
netflix_usfilms1 %>%
  as.data.frame() %>%
  mutate(k4 = factor(km_usfilms1$cluster)) %>%
  ggplot(data = ., aes(x = years_old , y = length, colour = k4)) + geom_point() + ggthemes::scal
e_color_colorblind() + facet_grid(~min_age_rating) + labs(title = "Scatterplot", subtitle = "vis
ualizing the 4 US films clusters", x = "Age", y= "Length")
```

Appendix 5. Scatterplot of netflix_nonusfilms1

```
netflix_nonusfilms1 %>%
  as.data.frame() %>%
  mutate(k4 = factor(km_nonusfilms1$cluster)) %>%
  ggplot(data = ., aes(x = years_old , y = length, colour = k4)) + geom_point() + ggthemes::scal
e_color_colorblind() + facet_grid(~min_age_rating) + labs(title = "Scatterplot", subtitle = "vis
ualizing the 4 non-US films clusters", x = "Age", y= "Length")
```


Scatterplot
visualizing the 4 non-US films clusters
.8750092461587
.5669742131115
.7968866304933
12721846864852
28123598517216
74328853474308
Length
Age
k4
1
2
3
4