STA 380 Part 2 Exercises (Report)

Alexander Schmelzeis

2024-08-18

Name: Alexander Schmelzeis
Assignment: Exercise 1 - Probability Practice
Date: August 7th, 2024

Part A: Visitors to your website are asked to answer a single survey question before they get access to the content on the page. Among all of the users, there are two categories: Random Clicker (RC), and Truthful Clicker (TC). There are two possible answers to the survey: yes and no. Random clickers would click either one with equal probability. You are also giving the information that the expected fraction of random clickers is 0.3. After a trial period, you get the following survey results: 65% said Yes and 35% said No. What fraction of people who are truthful clickers answered yes? Hint: use the rule of total probability.

To determine the fraction of people who answered yes given that they are truthful clickers, the rule of total probability must be used. In this instance, the formula would be total probability with regards to the probability of a yes, as follows:

P(Y) = P(Y|RC) * P(RC) + P(Y|TC) * P(TC)

Here, P(Y) is the probability of answering Yes, P(Y|RC) is the probability of a Yes given that they are a random clicker, P(RC) is the probability of someone being a random clicker, P(Y|TC) is the probability of a Yes given that they are a truthful clicker, and P(TC) is the probability of someone being a truthful clicker. As such, we want to find the value of P(Y|TC). This is done by first substituting the known probabilities which comprise the total probability of a yes.

P(Y) is exactly 0.65, based on the survey results. P(Y|RC) is 0.5, since random clickers click either yes or no with equal probability. P(RC) is 0.3, as this is the expected fraction of random clickers. This means that P(TC) is simply 0.7, as there are only two categories for users to fall under, and 1 - p(RC) is 0.7.

After gathering the known probabilities, P(Y|TC) must be isolated in the total probability formula. Substituting these known probabilities gives the following:

0.65 = 0.5 * 0.3 + P(Y|TC) * 0.7

0.65 = 0.15 + P(Y|TC) * 0.7

0.5 = P(Y|TC) * 0.7

P(Y|TC) = 0.5 / 0.7 = 0.7143

Therefore, the fraction of people who are truthful clickers which answered yes is roughly 0.7143 (71.43%).

Part B: Imagine a medical test for a disease with the following two attributes: - The sensitivity is about 0.993. That is, if someone has the disease, there is a probability of 0.993 that they will test positive. - The specificity is about 0.9999. This means that if someone doesn't have the disease, there is probability of 0.9999 that they will test negative. - In the general population, incidence of the disease is reasonably rare: about 0.0025% of all people have it (or 0.000025 as a decimal probability).

Suppose someone tests positive. What is the probability that they have the disease?

To determine the probability of someone having the disease given that they test positive, Bayes' Theorem must be utilized. In this particular problem, the formula would be with regards to the probability of disease given a positive test, as follows:

P(D|T+) = (P(T+|D) * P(D)) / P(T+)

Here, P(D|T+) is the probability of someone having the disease given that they test positive, P(T+|D) is the probability of testing positive given that they have the disease, P(D) is the probability of having the disease, and P(T+) is the probability of testing positive. As such, we want to find the value of P(D|T+). Like Part A, this is done by first substituting the known probabilities into the Bayes' Theorem formula.

P(T+|D) is provided in the problem already, with a probability of 0.993. P(D) is also given in the problem, with a probability of merely 0.000025. However P(T+) is not directly given in the problem. Because of this, the rule of total probability must be used to determine this value. The following formula illustrates how P(T+) would be calculated based on the rule of total probability:

P(T+) = P(T+|D) * P(D) + P(T+|No D) * P(No D)

In this case, P(T+) is the probability of testing positive, P(T+|D) is the probability of testing positive given that they have the disease, P(D) is the probability of having the disease, P(T+|No D) is the probability of testing positive given that they do not have the disease (false positive), and P(No D) is the probability of not having the disease. As stated before, P(T+|D) is 0.993 and P(D) is 0.000025. P(T+|No D) is simply 1 - P(T-|No D), or 1 minus the specificity of 0.9999. As such, P(T+|No D) would be around 0.0001. P(No D) is calculated similarly, as it would be 1 - P(D), or 1 minus the incidence of the disease of 0.000025. Therefore, P(No D) is approximately 0.999975. Plugging these values into the rule of total probability gives the following:

P(T+) = (0.993 * 0.000025) + (0.0001 * 0.999975)

P(T+) = 0.000024825 + 0.0000999975 = 0.0001248225

Now, we can substitute known probability values back into Bayes' Theorem with regards to P(D|T+). Solving for P(D|T+) results in the following:

P(D|T+) = (0.993 * 0.000025) / 0.0001248225

P(D|T+) = 0.000024825 / 0.0001248225 = 0.1989

Therefore, the probability that someone has the disease given that they test positive is roughly 0.1989 (19.89%).

Name: Alexander Schmelzeis
Assignment: Exercise 2 - Wrangling the Billboard Top 100
Date: August 8th, 2024

The responses to this exercise are located in the Jupyter notebook: "Wrangling the Billboard Top 100.ipynb" attached to the GitHub repository.

Name: Alexander Schmelzeis
Assignment: Exercise 3 - Visual Story Telling Part 1: Green Buildings
Date: August 10th, 2024

The responses to this exercise are located in the Jupyter notebook: "Visual Story Telling Part 1 - Green Buildings.ipynb" attached to the GitHub repository.

Name: Alexander Schmelzeis
Assignment: Exercise 4 - Visual Story Telling Part 2: Capital Metro Data
Date: August 11th, 2024

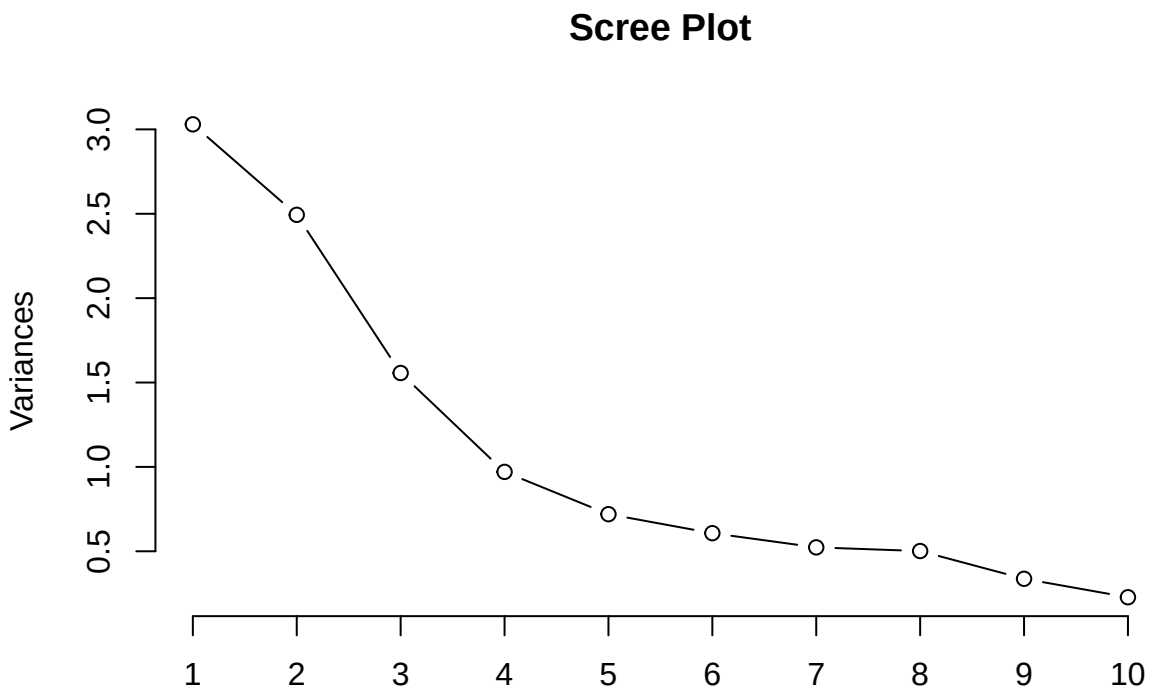
The responses to this exercise are located in the Jupyter notebook: "Visual Story Telling Part 2 - Capital Metro Data.ipynb" attached to the GitHub repository.

Name: Alexander Schmelzeis
Assignment: Exercise 5 - Clustering and Dimensionality Reduction
Date: August 12th, 2024

```
#PCA

library(ggplot2)
library(ggfortify)

# Loading the wine data
wine_data <- read.csv("C:\\Users\\aschm\\Downloads\\wine.csv")

# Selecting the 11 chemical properties
chemical_data <- wine_data[, c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar", "cl
                               "density", "pH", "sulphates", "alcohol")]

# Standardizing the data
chemical_data_scaled <- scale(chemical_data)

# Performing PCA
pca_result <- prcomp(chemical_data_scaled, center = TRUE, scale. = TRUE)

# Printing the PCA summary
summary(pca_result)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     1.7407 1.5792 1.2475 0.98517 0.84845 0.77930 0.72330
## Proportion of Variance 0.2754 0.2267 0.1415 0.08823 0.06544 0.05521 0.04756
## Cumulative Proportion  0.2754 0.5021 0.6436 0.73187 0.79732 0.85253 0.90009
##                           PC8     PC9   PC10    PC11
## Standard deviation     0.70817 0.58054 0.4772 0.18119
## Proportion of Variance 0.04559 0.03064 0.0207 0.00298
## Cumulative Proportion  0.94568 0.97632 0.9970 1.00000
```

```
# Visualizing the variance explained by each PC
screeplot(pca_result, type = "lines", main = "Scree Plot")
```




```
loadings <- pca_result$rotation

# Printing the contribution of each variable to the components
loadings
```

```
##                              PC1         PC2         PC3         PC4         PC5
## fixed.acidity        -0.23879890 -0.33635454 -0.43430130  0.16434621 -0.1474804
## volatile.acidity     -0.38075750 -0.11754972  0.30725942  0.21278489  0.1514560
## citric.acid           0.15238844 -0.18329940 -0.59056967 -0.26430031 -0.1553487
## residual.sugar        0.34591993 -0.32991418  0.16468843  0.16744301 -0.3533619
## chlorides            -0.29011259 -0.31525799  0.01667910 -0.24474386  0.6143911
## free.sulfur.dioxide   0.43091401 -0.07193260  0.13422395 -0.35727894  0.2235323
## total.sulfur.dioxide  0.48741806 -0.08726628  0.10746230 -0.20842014  0.1581336
## density              -0.04493664 -0.58403734  0.17560555  0.07272496 -0.3065613
## pH                   -0.21868644  0.15586900  0.45532412 -0.41455110 -0.4533764
## sulphates            -0.29413517 -0.19171577 -0.07004248 -0.64053571 -0.1365769
## alcohol              -0.10643712  0.46505769 -0.26110053 -0.10680270 -0.1888920
##                              PC6         PC7          PC8        PC9
## fixed.acidity        -0.20455371 -0.28307944  0.401235645 -0.3440567
## volatile.acidity     -0.49214307 -0.38915976 -0.087435088  0.4969327
## citric.acid           0.22763380 -0.38128504 -0.293412336  0.4026887
## residual.sugar       -0.23347775  0.21797554 -0.524872935 -0.1080032
## chlorides             0.16097639 -0.04606816 -0.471516850 -0.2964437
## free.sulfur.dioxide  -0.34005140 -0.29936325  0.207807585 -0.3666563
## total.sulfur.dioxide -0.15127722 -0.13891032  0.128621319  0.3206955
```

```
## density               0.01874307 -0.04675897  0.004831136 -0.1128800
## pH                    0.29657890 -0.41890702 -0.028643277 -0.1278367
## sulphates            -0.29692579  0.52534311  0.165818022  0.2077642
## alcohol              -0.51837780 -0.10410343 -0.399233887 -0.2518903
##                             PC10          PC11
## fixed.acidity         0.281267685  0.3346792663
## volatile.acidity     -0.152176731  0.0847718098
## citric.acid          -0.234463340 -0.0011089514
## residual.sugar        0.001372773  0.4497650778
## chlorides             0.196630217  0.0434375867
## free.sulfur.dioxide  -0.480243340 -0.0002125351
## total.sulfur.dioxide  0.713663486 -0.0626848131
## density               0.003908289 -0.7151620723
## pH                    0.141310977  0.2063605036
## sulphates            -0.045959499  0.0772024671
## alcohol               0.205053085 -0.3357018784
```

```
# Plotting the PC1 and PC2
autoplot(pca_result, data = wine_data, colour = 'color')
```

After running PCA on the wine data, some interesting results emerge. When examining the proportion of variance under the importance of components, it can be observed that PC1 explains about 27.54% of the variance in the data set, with a standard deviation of roughly 1.7407. PC2 explains roughly 22.67% of the variance and PC3 explains about 14.15% of the variance, with standard deviations of 1.5792 and 1.2475, respectively. This means that all together, the first three principal components explain a majority of the variation in the data set, adding up to about 64.36% of the total variance. PC4 through PC8 only

contributes slightly when concerning proportion of variance, with ranges of variance explained form 8.82% to 4.56%. The last three principal components only explain a very small portion of the variance collectively, even as little as 0.3% (PC11).

When looking at the principal components weights for these variables, several insights can be derived from the first three principal components. In PC1, the variable weights with the highest absolute values are total sulfur dioxide (0.487), free sulfur dioxide (0.431), and residual sugar (0.346). This indicates that these chemical properties have the strongest influence with regards to PC1. In turn, they help the principal component to explain the largest amount of variance in the wine data. For PC2, the variable weights with the greatest absolute values are alcohol (0.465), density (-0.584), and fixed acidity (-0.336), suggesting that they have the strongest influence on PC2. This also reveals that this component might have something to do with an association between alcohol and density as it relates to wines. For PC3, the three primary contributors by variable weight absolute value are citric acid (-0.591), pH (0.455), and fixed acidity (-0.434). This highlights the fact that this component is possibly related to chemical properties of wines related to acidity.

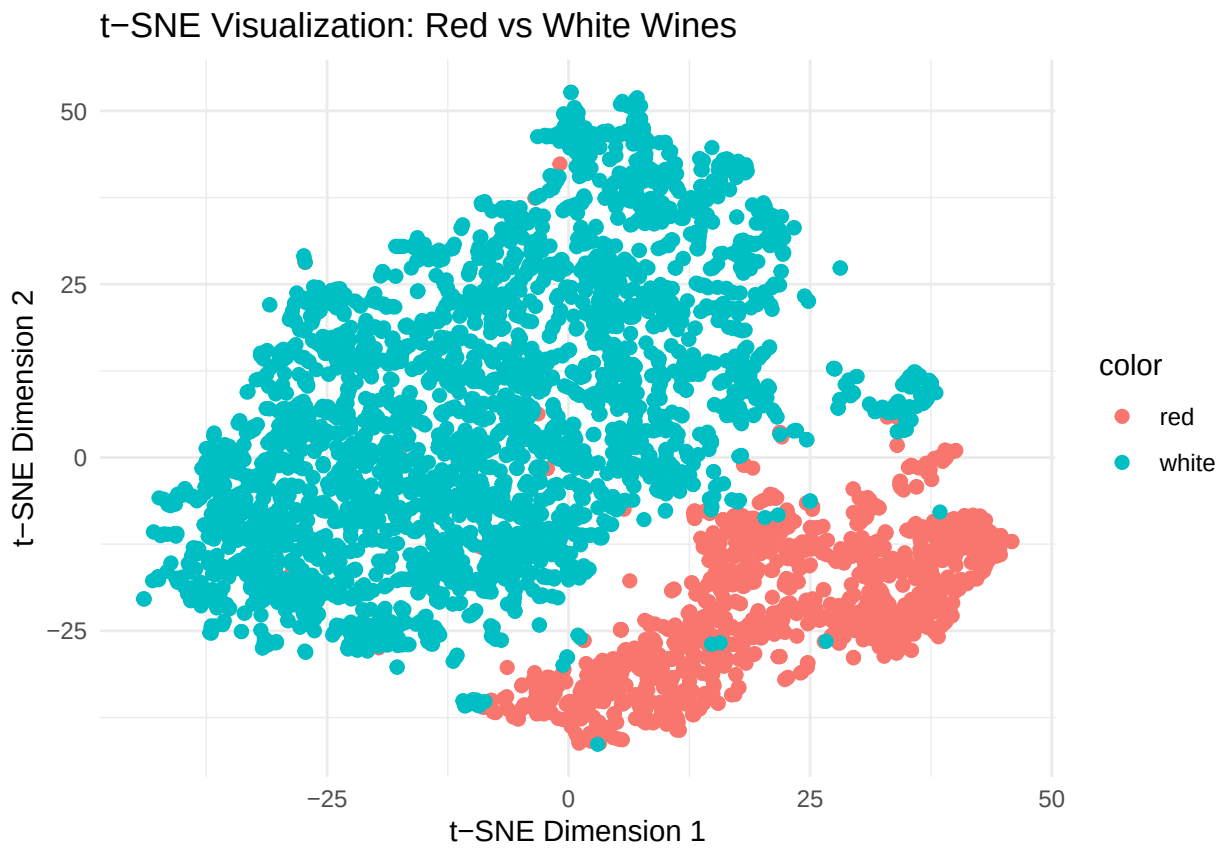
From the principal components plot of the first two principal components, it can be determined that PCA does a fairly decent job of distinguishing between red and white wines. For the most part, there appears to be a clear separation between what is determined to be red wine and white wine, even when only considering two principal components. However, because there are still some points which overlap with regards to the classification of wine, this dimensionality reduction technique is probably not the best for this data set.

```
#tSNE

library(Rtsne)
library(ggplot2)

# Loading the wine data
wine_data <- read.csv("C:\\Users\\aschm\\Downloads\\wine.csv")

# Selecting the 11 chemical properties
chemical_data <- wine_data[, c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar", "cl
                               "density", "pH", "sulphates", "alcohol")]

# Standardizing the data
chemical_data_scaled <- scale(chemical_data)

# Removing duplicate rows
chemical_data_scaled_unique <- unique(chemical_data_scaled)

# Running t-SNE
set.seed(42)
tsne_result <- Rtsne(chemical_data_scaled_unique, perplexity = 30, dims = 2, verbose = TRUE)
```

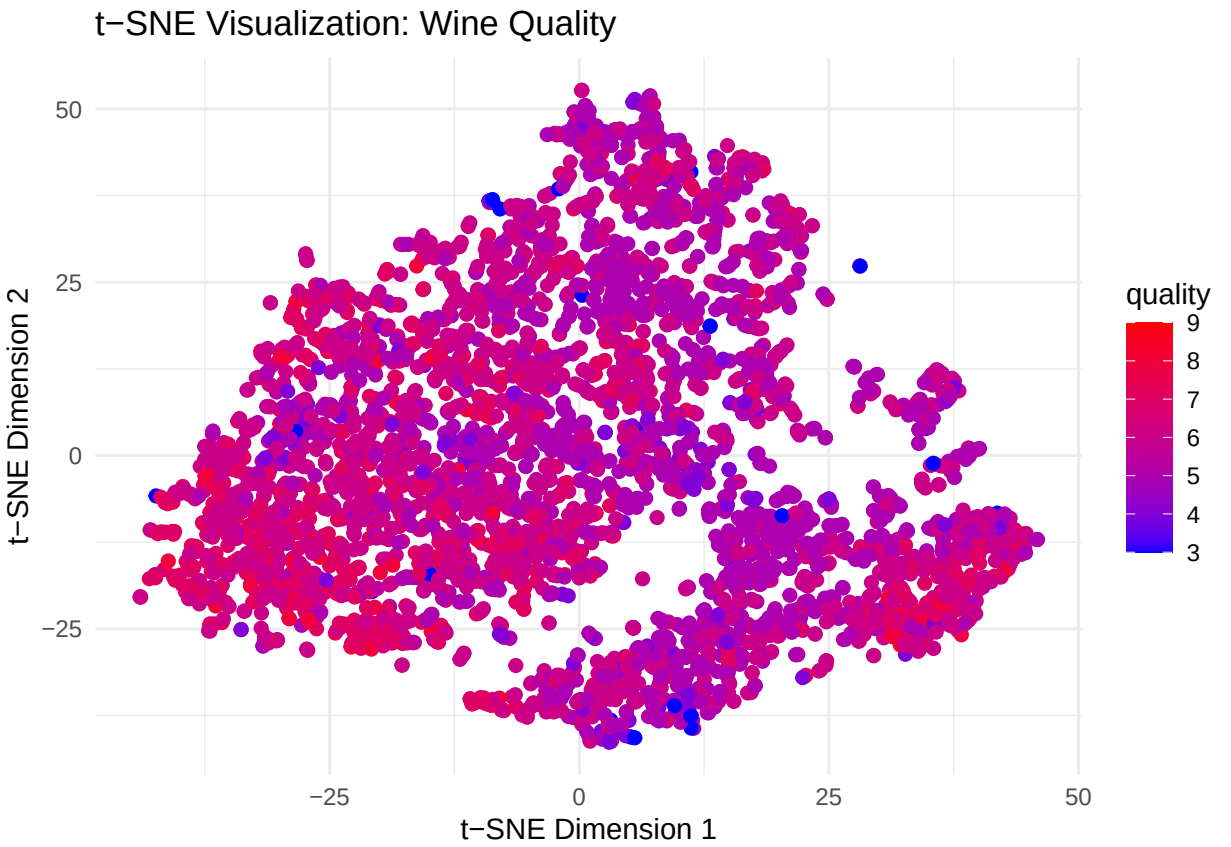
```
## Performing PCA
## Read the 5318 x 11 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.96 seconds (sparsity = 0.024032)!
## Learning embedding...
## Iteration 50: error is 89.829866 (50 iterations in 0.59 seconds)
## Iteration 100: error is 81.783766 (50 iterations in 0.60 seconds)
## Iteration 150: error is 81.031047 (50 iterations in 0.55 seconds)
```

```
## Iteration 200: error is 80.941518 (50 iterations in 0.53 seconds)
## Iteration 250: error is 80.904879 (50 iterations in 0.52 seconds)
## Iteration 300: error is 2.665579 (50 iterations in 0.53 seconds)
## Iteration 350: error is 2.293344 (50 iterations in 0.50 seconds)
## Iteration 400: error is 2.098616 (50 iterations in 0.53 seconds)
## Iteration 450: error is 1.977251 (50 iterations in 0.52 seconds)
## Iteration 500: error is 1.894778 (50 iterations in 0.54 seconds)
## Iteration 550: error is 1.837061 (50 iterations in 0.52 seconds)
## Iteration 600: error is 1.797778 (50 iterations in 0.52 seconds)
## Iteration 650: error is 1.769071 (50 iterations in 0.52 seconds)
## Iteration 700: error is 1.749839 (50 iterations in 0.53 seconds)
## Iteration 750: error is 1.734547 (50 iterations in 0.53 seconds)
## Iteration 800: error is 1.722583 (50 iterations in 0.53 seconds)
## Iteration 850: error is 1.712501 (50 iterations in 0.53 seconds)
## Iteration 900: error is 1.703988 (50 iterations in 0.53 seconds)
## Iteration 950: error is 1.697304 (50 iterations in 0.53 seconds)
## Iteration 1000: error is 1.692148 (50 iterations in 0.53 seconds)
## Fitting performed in 10.67 seconds.
```

```r
# Extracting the t-SNE coordinates
tsne_coords <- as.data.frame(tsne_result$Y)

# Adding the color and quality labels for visualization
wine_data_unique <- wine_data[!duplicated(chemical_data_scaled), ]
tsne_coords$color <- wine_data_unique$color
tsne_coords$quality <- wine_data_unique$quality

# Plotting the t-SNE results by wine color
ggplot(tsne_coords, aes(x = V1, y = V2, color = color)) +
  geom_point(size = 2) +
  labs(title = "t-SNE Visualization: Red vs White Wines",
       x = "t-SNE Dimension 1",
       y = "t-SNE Dimension 2") +
  theme_minimal()
```




```
# Plotting the t-SNE results by wine quality
ggplot(tsne_coords, aes(x = V1, y = V2, color = quality)) +
  geom_point(size = 2) +
  scale_color_gradient(low = "blue", high = "red") +
  labs(title = "t-SNE Visualization: Wine Quality",
       x = "t-SNE Dimension 1",
       y = "t-SNE Dimension 2") +
  theme_minimal()
```



When running tSNE on the wine data, I first had to standardize the chemical properties in the data via the scale() function. This is due to the fact that this dimensionality reduction techniques performs considerably better when working with standardized as opposed to unstandardized data. After that, duplicate rows were removed, and the Rtsne() funciton was utilized to run tSNE. The perplexity parameter was set to 30, while the dims parameter was set to 2. This is because I wanted to reduce the dimensions of the data to 2, for the purposes of easier visualization. Furthermore, ggplot() was used to plot the tSNE results by both wine color and wine quality. In the wine color visualization, the colors of each point represent the color of the white (either white or red wine). In the wine quality visualization, the colors of each point represent the quality of the wine (anywhere from 1 to 10).

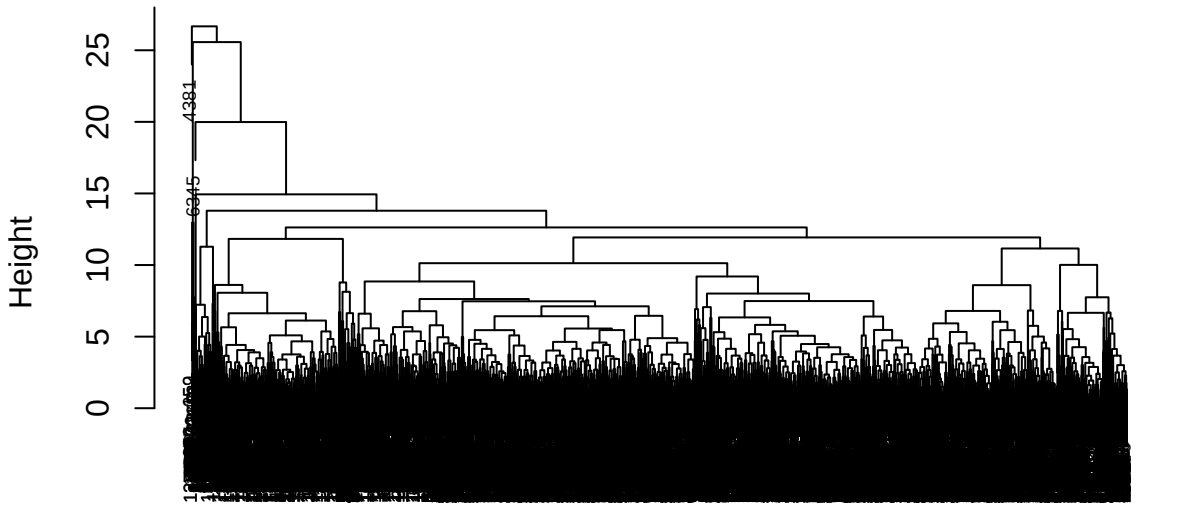
Among all of the dimensionality reduction techniques used in this problem, tSNE make the most sense for me as it concerns the wine data. This is primarily due to the fact that tSNE is inherently non-linear, which allows it to capture relationships between chemical properties that are not necessarily linear in nature. It is also highly effective when it comes to the preservation of local relationships of the data when dealing with lower dimensions. Both of these aspects of tSNE make it critical when potentially distinguishing between red or white wines, or even the deemed qualities of different wines.

When looking at the first tSNE visualization by wine color, there seems to be a considerable separation between red and white wines in the two dimensional space. Although a few points are misclassified (some are distinguished incorrectly), it is not enough for the clusters to actually be overlapping one another. This clear separation between the two types of wines suggests that just the chemical properties of wines alone are sufficient enough to classify a wine as either red or white. Given the tSNE coordinates of each cluster in the 2-dimensional space, it is safe to conclude that tSNE is easily capable of distinguishing between the reds from the whites.

However, when looking at the tSNE visualization by wine quality, tSNE does not seem capable of distinguishing the higher quality wines from the lower quality wines. Contrary to the wine color visualization, there seems to be no structure in the plot, with wines of varying qualities being clumped together and considered

to be similar. The clusters seem to be based on something independent from the quality, since there is no gradient or clustering relating to the different quality scores. Because there is no obvious pattern or structure in the visualization, it suggests that the chemical properties of the wines alone might not be sufficient in effectively distinguishing wines on the basis of wine quality, at least in an unsupervised fashion. As such, a supervised technique is likely needed in order to effectively classify the qualities of wines properly.

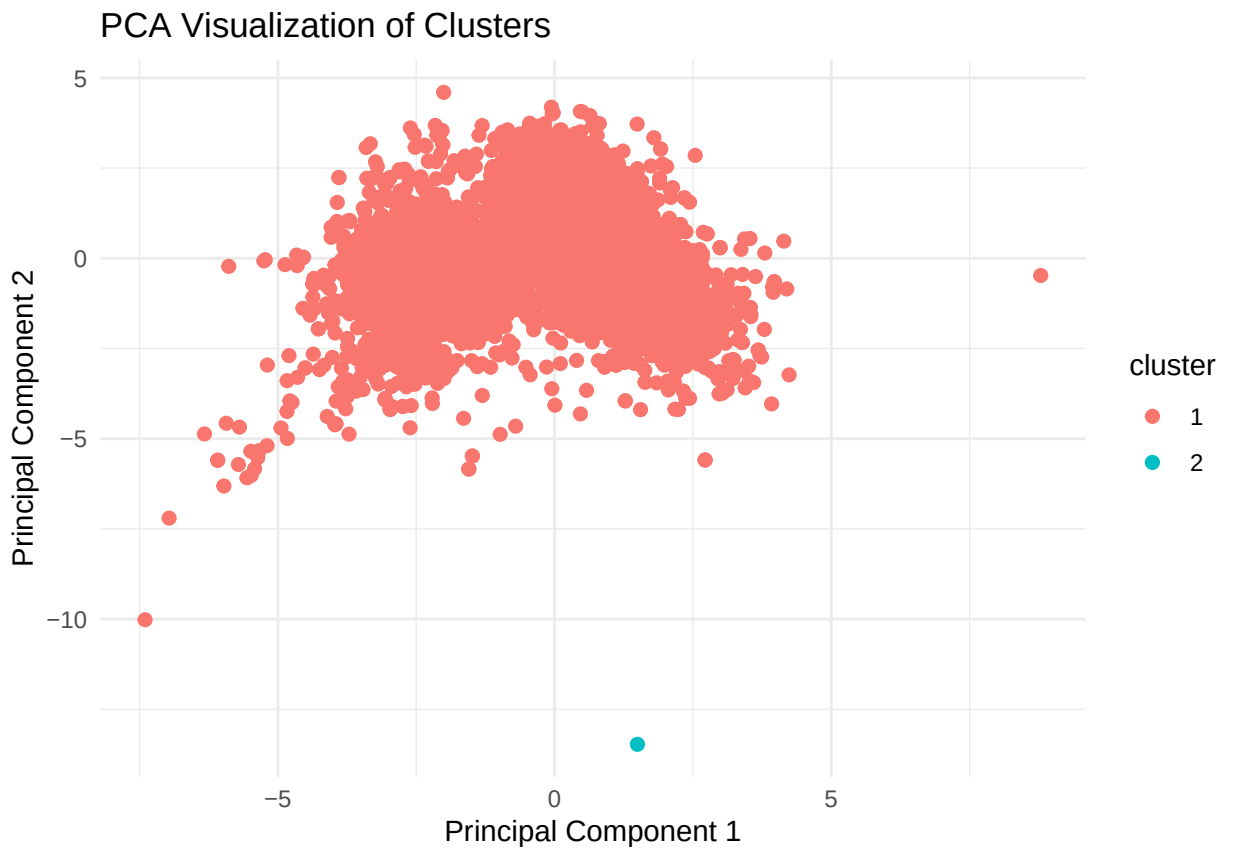
```
#Hierarchical Clustering

library(ggplot2)

# Loading the wine data
wine_data <- read.csv("C:\\Users\\aschm\\Downloads\\wine.csv")

# Selecting the 11 chemical properties
chemical_data <- wine_data[, c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar", "cl
                               "density", "pH", "sulphates", "alcohol")]

# Standardizing the data
chemical_data_scaled <- scale(chemical_data)

# Computing the distance matrix
distance_matrix <- dist(chemical_data_scaled, method = "euclidean")

# Performing hierarchical clustering
hc_complete <- hclust(distance_matrix, method = "complete")

# Plotting the dendrogram
plot(hc_complete, main = "Dendrogram of Hierarchical Clustering (Complete Linkage)",
     xlab = "", sub = "", cex = 0.6)
```




```r
clusters <- cutree(hc_complete, k = 2)

# Adding the clusters to the original data
wine_data$cluster <- as.factor(clusters)

# Summarizing the clusters
table(wine_data$cluster)
```

```
##
##    1    2
## 6496    1
```

```r
# Visualizing the clusters with the first two principal components
pca_result <- prcomp(chemical_data_scaled, center = TRUE, scale. = TRUE)
pca_data <- as.data.frame(pca_result$x[, 1:2])
pca_data$cluster <- wine_data$cluster

# Plotting the clusters
ggplot(pca_data, aes(x = PC1, y = PC2, color = cluster)) +
  geom_point(size = 2) +
  labs(title = "PCA Visualization of Clusters",
       x = "Principal Component 1",
       y = "Principal Component 2") +
  theme_minimal()
```



The clustering algorithm I chose to run was hierarchical clustering, although this proved to be extremely ineffective of making any kind of distinctions. When running hierarchical clustering, I first selected the eleven chemical properties from the wine data. Because the performance of clustering improves when working with standardized data as opposed to unstandardized data, I standardized the chemical properties using the scale() function. Then, I computed a distance matrix using Euclidean distance and performed hierarchical clustering with complete linkage. This was done using R's hclust() function. After that, the dendrogram was plotted, and the clusters were added to the original data. When visualizing the resulting clusters, I first performed PCA, such that they can be plotted in a two-dimensional space. The function ggplot() was then utilized in actualizing a plot of the clusters in the PCA space, producing the final PCA cluster visualization. When creating two clusters (representing red and white wines), the results were insufficient. After cutting the dendrogram to form two clusters, the first cluster contained 6496 data points, while the second cluster only contained 1 data point. As such, there is basically no distinguishing ability of hierarchical clustering for the wine data set, at least as it concerns two clusters. Evidently, it is incapable of distinguishing wines on the basis of either wine color or wine quality.

Name: Alexander Schmelzeis
Assignment: Exercise 6 - Market Segmentation
Date: August 14th, 2024

```r
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(ggplot2)
library(cluster)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(reshape2)

# Loading the dataset
social_data <- read.csv("C:\\Users\\aschm\\Downloads\\social_marketing.csv")

# Inspecting the dataset structure
str(social_data)
```

```
## 'data.frame':    7882 obs. of  37 variables:
##  $ X                : chr  "hmjoe4g3k" "clk1m5w8s" "jcsovtak3" "3oeb4hiln" ...
##  $ chatter          : int  2 3 6 1 5 6 1 5 6 5 ...
##  $ current_events   : int  0 3 3 5 2 4 2 3 2 2 ...
##  $ travel           : int  2 2 4 2 0 2 7 3 0 4 ...
##  $ photo_sharing    : int  2 1 3 2 6 7 1 6 1 4 ...
##  $ uncategorized    : int  2 1 1 0 1 0 0 1 0 0 ...
##  $ tv_film          : int  1 1 5 1 0 1 1 1 0 5 ...
##  $ sports_fandom    : int  1 4 0 0 0 1 1 1 0 9 ...
##  $ politics         : int  0 1 2 1 2 0 11 0 0 1 ...
##  $ food             : int  4 2 1 0 0 2 1 0 2 5 ...
##  $ family           : int  1 2 1 1 1 1 0 0 2 4 ...
##  $ home_and_garden  : int  2 1 1 0 0 1 0 0 1 0 ...
##  $ music            : int  0 0 1 0 0 1 0 2 1 1 ...
##  $ news             : int  0 0 1 0 0 0 1 0 0 0 ...
##  $ online_gaming    : int  0 0 0 0 3 0 0 1 2 1 ...
##  $ shopping         : int  1 0 2 0 2 5 1 3 0 0 ...
##  $ health_nutrition: int  17 0 0 0 0 0 1 1 22 7 ...
##  $ college_uni      : int  0 0 0 1 4 0 1 0 1 4 ...
##  $ sports_playing   : int  2 1 0 0 0 0 1 0 0 1 ...
##  $ cooking          : int  5 0 2 0 1 0 1 10 5 4 ...
##  $ eco              : int  1 0 1 0 0 0 0 0 2 1 ...
##  $ computers        : int  1 0 0 0 1 1 1 1 1 2 ...
##  $ business         : int  0 1 0 1 0 1 3 0 1 0 ...
##  $ outdoors         : int  2 0 0 0 1 0 1 0 3 0 ...
##  $ crafts           : int  1 2 2 3 0 0 0 1 0 0 ...
```

```
##  $ automotive      : int  0 0 0 0 0 1 0 1 0 4 ...
##  $ art             : int  0 0 8 2 0 0 1 0 1 0 ...
##  $ religion        : int  1 0 0 0 0 0 1 0 0 13 ...
##  $ beauty          : int  0 0 1 1 0 0 0 5 5 1 ...
##  $ parenting       : int  1 0 0 0 0 0 0 1 0 3 ...
##  $ dating          : int  1 1 1 0 0 0 0 0 0 0 ...
##  $ school          : int  0 4 0 0 0 0 0 0 1 3 ...
##  $ personal_fitness: int  11 0 0 0 0 0 0 0 12 2 ...
##  $ fashion         : int  0 0 1 0 0 0 0 4 3 1 ...
##  $ small_business  : int  0 0 0 0 1 0 0 0 1 0 ...
##  $ spam            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ adult           : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# Removing the first column (X (User ID))
social_data <- social_data[, -1]

# Checking for missing values
sum(is.na(social_data))
```

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

```
# Normalizing the data
social_data_scaled <- as.data.frame(scale(social_data))
```

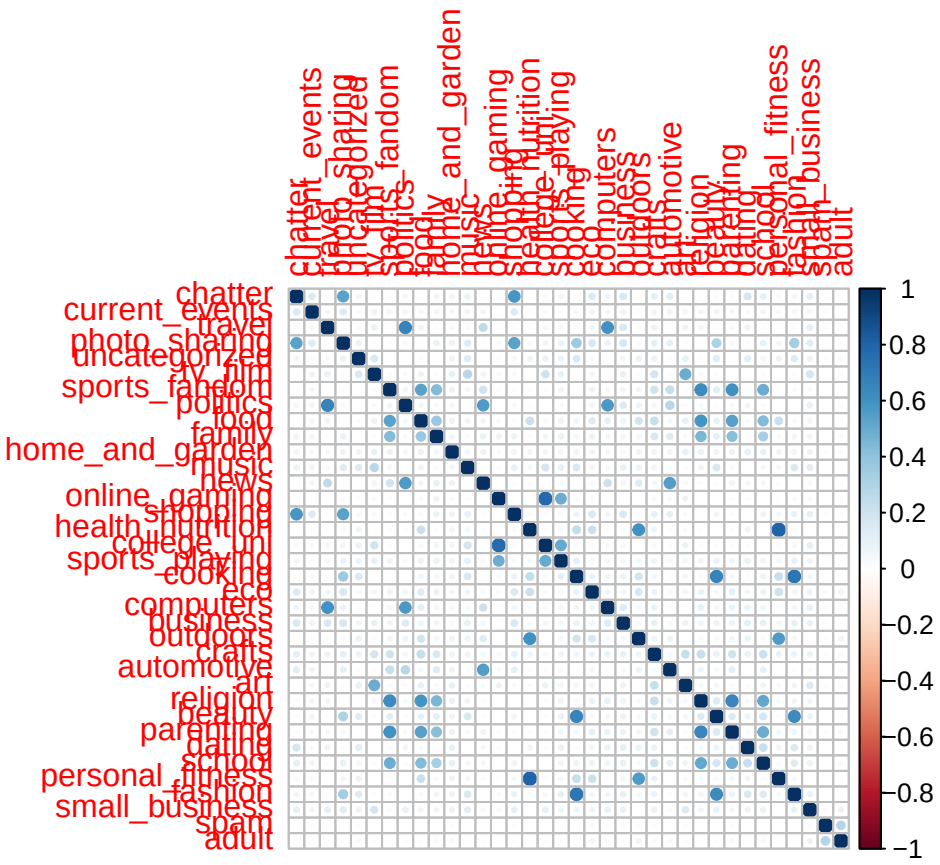
When analyzing the social marketing data, the first course of action was to pre-process the data. To accomplish this task, I first examined the structure of the data. As it turns out, all of the columns but the column pertaining to the user identification were of the int data type; the user identification column was of the chr type. Because it is the only non-numerical column in the dataset, I did not feel as if it was crucial for the overall analysis of the data. As such, it was promptly removed from the dataset. After checking the data for missing values, of which there were none, I normalized the data, in order to allow for more effective dimensionality reduction later on. This was done by first using the scale() function on the data set, then utilizing the as.data.frame() function to store the results in a data frame. This normalized data was stored in the variable social_data_scaled.

```
summary(social_data)
```

```
##     chatter       current_events      travel       photo_sharing
##  Min.   : 0.000   Min.   :0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 2.000   1st Qu.:1.000   1st Qu.: 0.000   1st Qu.: 1.000
##  Median : 3.000   Median :1.000   Median : 1.000   Median : 2.000
##  Mean   : 4.399   Mean   :1.526   Mean   : 1.585   Mean   : 2.697
##  3rd Qu.: 6.000   3rd Qu.:2.000   3rd Qu.: 2.000   3rd Qu.: 4.000
##  Max.   :26.000   Max.   :8.000   Max.   :26.000   Max.   :21.000
##  uncategorized      tv_film      sports_fandom       politics
##  Min.   :0.000   Min.   : 0.00   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:0.000   1st Qu.: 0.00   1st Qu.: 0.000   1st Qu.: 0.000
##  Median :1.000   Median : 1.00   Median : 1.000   Median : 1.000
##  Mean   :0.813   Mean   : 1.07   Mean   : 1.594   Mean   : 1.789
##  3rd Qu.:1.000   3rd Qu.: 1.00   3rd Qu.: 2.000   3rd Qu.: 2.000
##  Max.   :9.000   Max.   :17.00   Max.   :20.000   Max.   :37.000
##       food            family        home_and_garden      music
##  Min.   : 0.000   Min.   : 0.0000   Min.   :0.0000   Min.   : 0.0000
```

```
##  1st Qu.: 0.000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.0000
##  Median : 1.000   Median : 1.0000   Median :0.0000   Median : 0.0000
##  Mean   : 1.397   Mean   : 0.8639   Mean   :0.5207   Mean   : 0.6793
##  3rd Qu.: 2.000   3rd Qu.: 1.0000   3rd Qu.:1.0000   3rd Qu.: 1.0000
##  Max.   :16.000   Max.   :10.0000   Max.   :5.0000   Max.   :13.0000
##       news        online_gaming       shopping      health_nutrition
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000
##  Median : 0.000   Median : 0.000   Median : 1.000   Median : 1.000
##  Mean   : 1.206   Mean   : 1.209   Mean   : 1.389   Mean   : 2.567
##  3rd Qu.: 1.000   3rd Qu.: 1.000   3rd Qu.: 2.000   3rd Qu.: 3.000
##  Max.   :20.000   Max.   :27.000   Max.   :12.000   Max.   :41.000
##   college_uni     sports_playing      cooking            eco
##  Min.   : 0.000   Min.   :0.0000   Min.   : 0.000   Min.   :0.0000
##  1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.: 0.000   1st Qu.:0.0000
##  Median : 1.000   Median :0.0000   Median : 1.000   Median :0.0000
##  Mean   : 1.549   Mean   :0.6392   Mean   : 1.998   Mean   :0.5123
##  3rd Qu.: 2.000   3rd Qu.:1.0000   3rd Qu.: 2.000   3rd Qu.:1.0000
##  Max.   :30.000   Max.   :8.0000   Max.   :33.000   Max.   :6.0000
##    computers          business         outdoors          crafts
##  Min.   : 0.0000   Min.   :0.0000   Min.   : 0.0000   Min.   :0.0000
##  1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.0000   1st Qu.:0.0000
##  Median : 0.0000   Median :0.0000   Median : 0.0000   Median :0.0000
##  Mean   : 0.6491   Mean   :0.4232   Mean   : 0.7827   Mean   :0.5159
##  3rd Qu.: 1.0000   3rd Qu.:1.0000   3rd Qu.: 1.0000   3rd Qu.:1.0000
##  Max.   :16.0000   Max.   :6.0000   Max.   :12.0000   Max.   :7.0000
##    automotive           art             religion          beauty
##  Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.000   Min.   : 0.0000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.: 0.0000
##  Median : 0.0000   Median : 0.0000   Median : 0.000   Median : 0.0000
##  Mean   : 0.8299   Mean   : 0.7248   Mean   : 1.095   Mean   : 0.7052
##  3rd Qu.: 1.0000   3rd Qu.: 1.0000   3rd Qu.: 1.000   3rd Qu.: 1.0000
##  Max.   :13.0000   Max.   :18.0000   Max.   :20.000   Max.   :14.0000
##    parenting           dating            school        personal_fitness
##  Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.000
##  Median : 0.0000   Median : 0.0000   Median : 0.0000   Median : 0.000
##  Mean   : 0.9213   Mean   : 0.7109   Mean   : 0.7677   Mean   : 1.462
##  3rd Qu.: 1.0000   3rd Qu.: 1.0000   3rd Qu.: 1.0000   3rd Qu.: 2.000
##  Max.   :14.0000   Max.   :24.0000   Max.   :11.0000   Max.   :19.000
##     fashion        small_business        spam              adult
##  Min.   : 0.0000   Min.   :0.0000   Min.   :0.00000   Min.   : 0.0000
##  1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 0.0000
##  Median : 0.0000   Median :0.0000   Median :0.00000   Median : 0.0000
##  Mean   : 0.9966   Mean   :0.3363   Mean   :0.00647   Mean   : 0.4033
##  3rd Qu.: 1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.: 0.0000
##  Max.   :18.0000   Max.   :6.0000   Max.   :2.00000   Max.   :26.0000
```

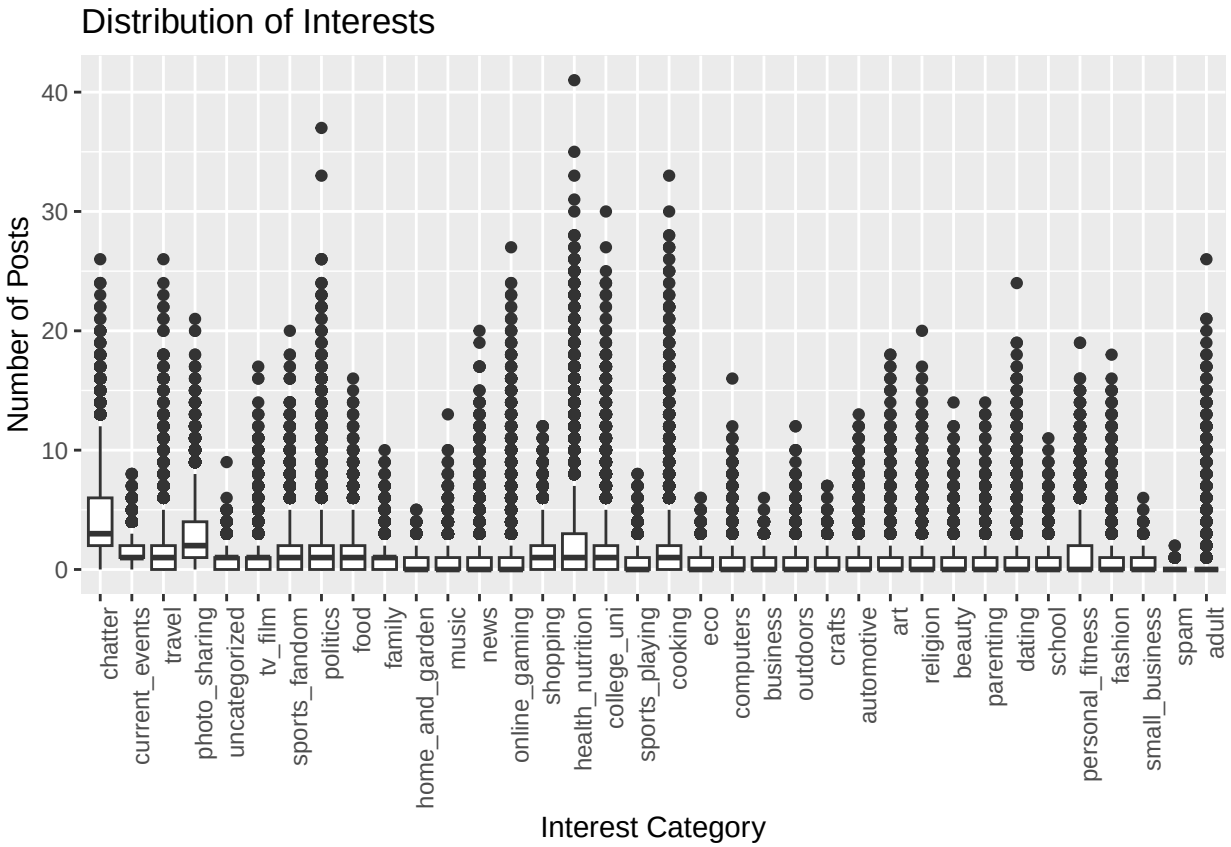
```r
# Correlation matrix
corr_matrix <- cor(social_data_scaled)
corrplot(corr_matrix, method = "circle")
```




```
# Visualizing the distribution of interests
melted_data <- melt(social_data)
```

```
## No id variables; using all as measure variables
```

```
ggplot(melted_data, aes(x = variable, y = value)) +
  geom_boxplot() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Distribution of Interests", x = "Interest Category", y = "Number of Posts")
```

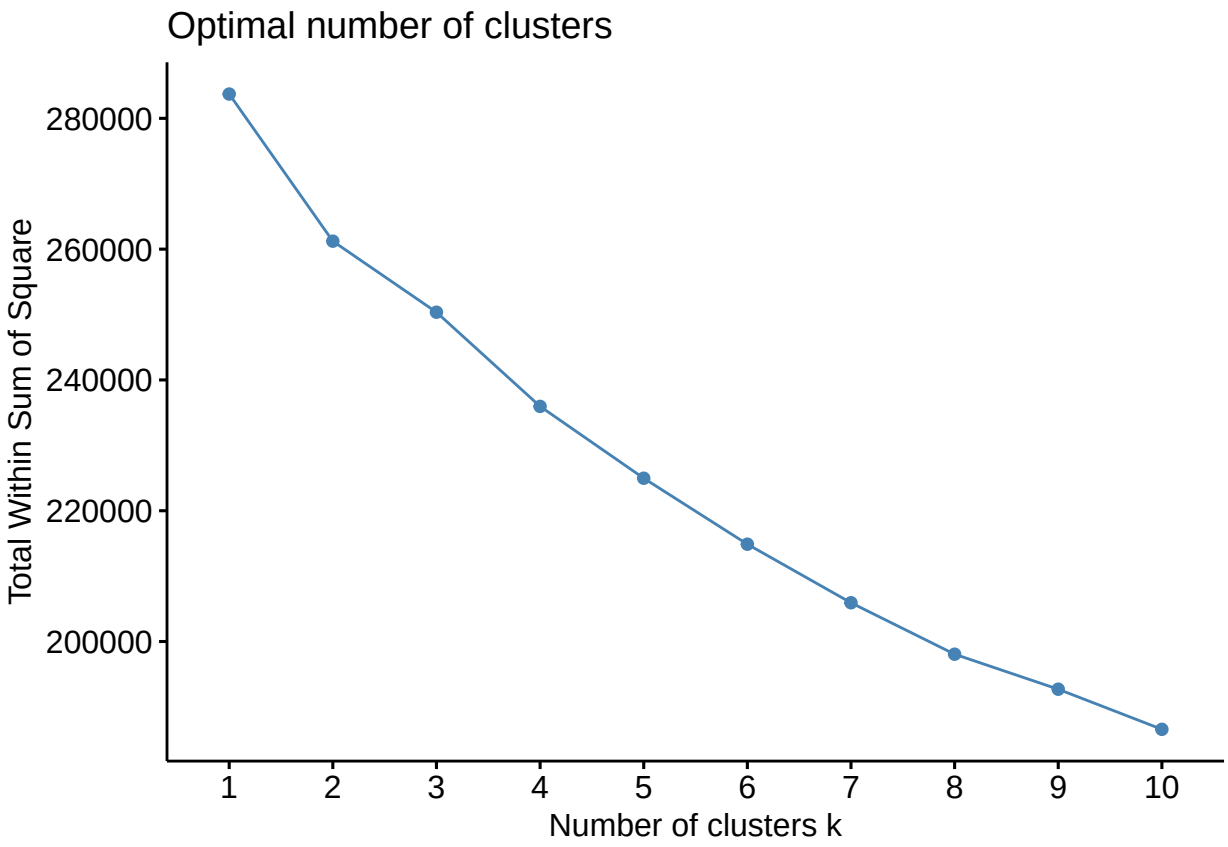


After pre-processing the data, I decided to perform some exploratory data analysis to see if any initial insights or patterns could be observed regarding the data. There were two plots developed via EDA which were of interest to me: a correlation matrix and a series of box plots encompassing all the different interest categories. The correlation matrix was derived first through the use of the cor() function of the scaled social media data. Then, corrplot() was used to generate the above correlation matrix, with the colors of circles pertaining to the strengths of correlations. When examining the matrix, there are several noticeable associations between interest categories. However, there are three such associations which seem to tower above the rest. For one, fashion and cooking appear to possess one of the strongest bivariate correlations, with a dark blue circle indicating a fairly significant positive correlation between the two interest categories. This suggests that people who enjoy all the aspects of the fashion industry are perhaps more likely to also be interested in cooking, which might give rise to a possible market segment to explore. A second notable association is between personal_fitness and health_nutrition, also possessing a darker blue circle. This signifies a fairly strong correlation between them, indicating that those users who are interested in personal fitness are also likely to be interested in health and nutrition. The association here seems to be slightly more intuitive, as people who exercise frequently are typically mindful of what they put into their bodies. That being said, it could still lead to an intriguing market segment to hone on. Lastly, a third significant association can be found between college_uni and online_gaming. As was the case with the other two correlations, the notable positive association between the two interest categories is marked by a darkish blue circle. This likely suggests that college students are potentially more inclined to partake in online gaming. Although this might seem surprising at first, this correlation is also quite intuitive. Since the online gaming space is dominated by younger people, it makes sense that college students (who are mainly young adults) would have interest in such a topic. The fact that both of these interests are common among a certain demographic of people could indicate another potential market segment to dissect.

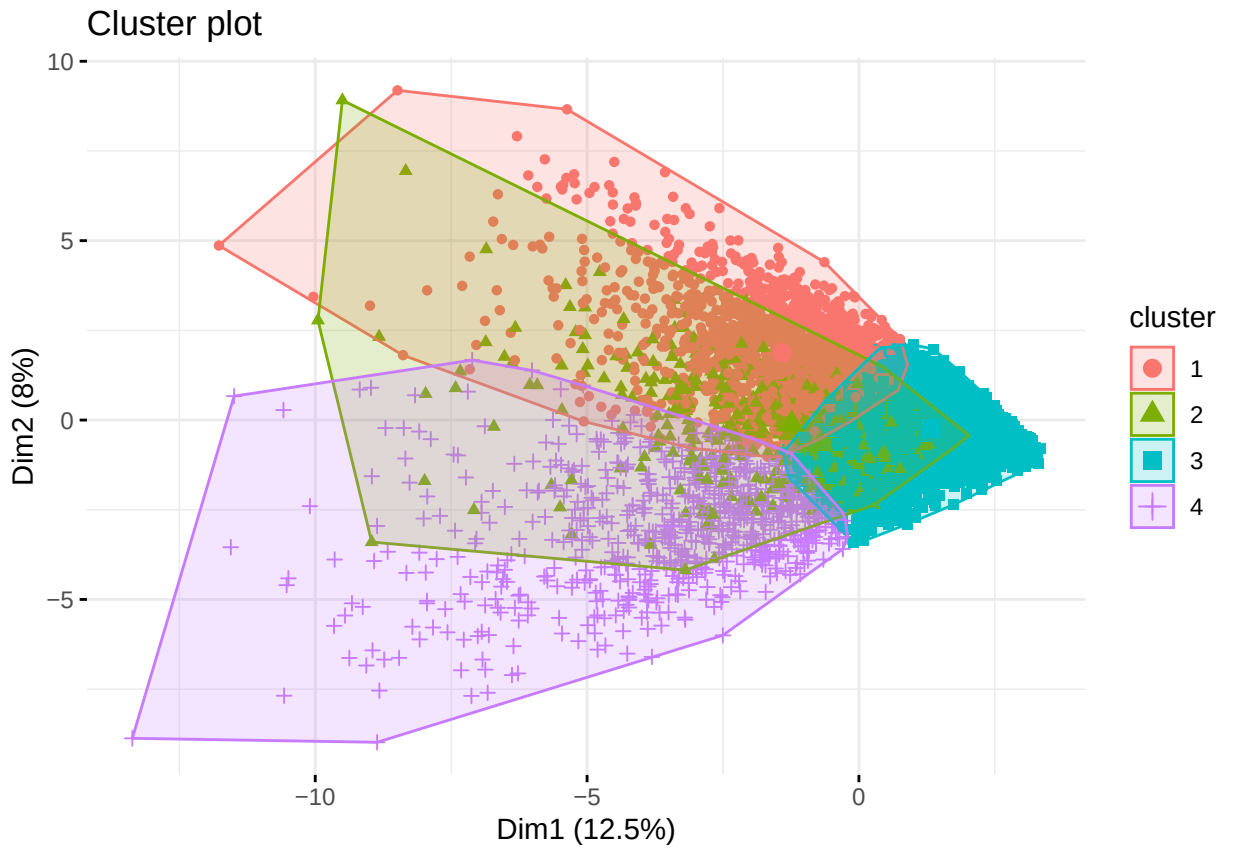
When looking at the series of box plots, interesting details about each of the interest categories emerge. First off, the ranges of the number of posts for each interest category vary quite dramatically, with every category possessing a considerable quantity of outliers. That being said, certain categories have much wider ranges

compared to others in the social marketing data. For instance, politics, health_nutrition and cooking have the widest ranges among interest categories, with some users posting as often as 40 times, as is shown in the health_nutrition category. Despite the differing ranges of each interest category, most categories seem to have a median number of posts close to 0 (as is seen in art and business). These low medians are accompanied by lower interquartile ranges, suggesting that a large portion of users are not actively engaged in most interest categories. Furthermore, it appears that some of the categories themselves possess an overall low level of engagement. This can be observed in categories with a smaller range in number of posts, as is the case with the eco, small_business, and home_and_garden categories. All of these interests have ranges of less than 10 posts. All in all, the box plots suggest that users typically have only a few overarching interests, where a majority of their social media posts relate to. Despite this, I believe that it is still worth analyzing the categories with supposedly low engagement, in order to identify more complete and comprehensive market segments.

```
# Determining the optimal number of clusters
fviz_nbclust(social_data_scaled, kmeans, method = "wss")
```



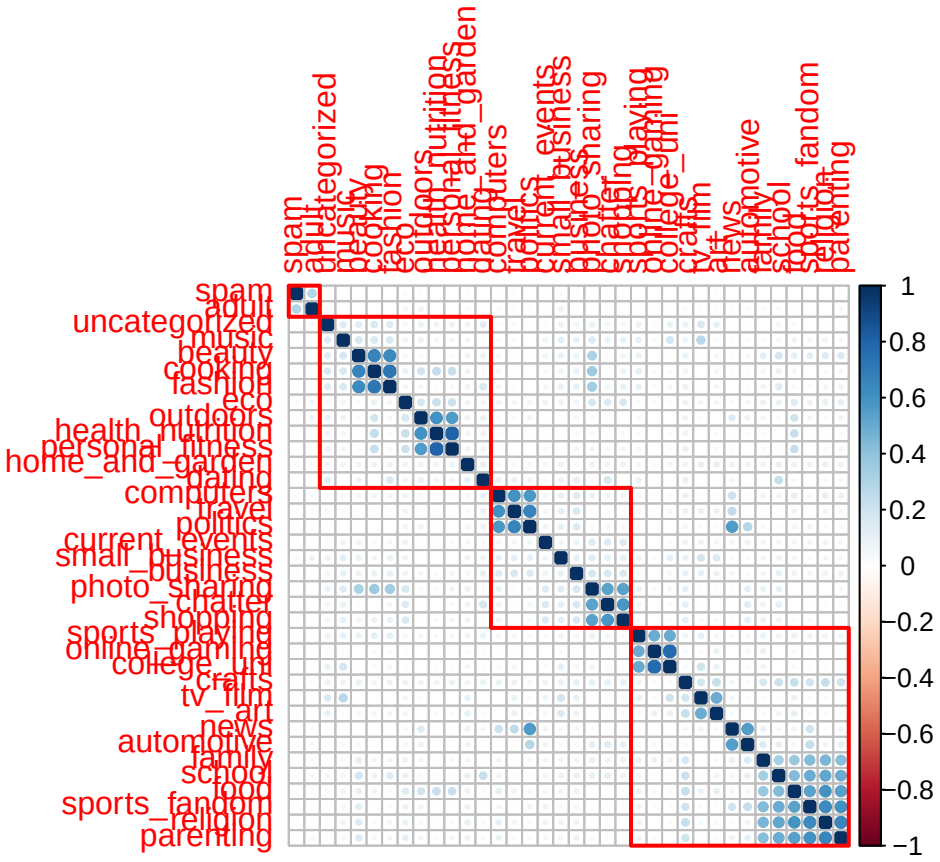

```
# Performing K-means clustering
set.seed(123)
kmeans_result <- kmeans(social_data_scaled, centers = 4, nstart = 25)

# Adding cluster assignments to the data
social_data$cluster <- as.factor(kmeans_result$cluster)

# Visualizing clusters
fviz_cluster(kmeans_result, data = social_data_scaled, geom = "point",
             ellipse.type = "convex", ggtheme = theme_minimal())
```



After the exploratory data analysis was completed, I opted to perform some general segmentation analysis using K-means clustering. This was done with the hope of discovering clear and distinct market segments based on the interest categories of social media users. The first course of action was to determine the optimal clusters for K-means clustering based on the scaled dataset. This was accomplished through the use of an elbow plot, through the fviz_nbclust() function in R, with the parameters of kmeans and a method of "wss" (within sum of square). For this case, the optimal number of clusters was deemed to be 4, as this is the point where the decrease in WSS becomes gradual, albeit not by much. As such there will be 4 associated market segments for NutrientH20 to possibly hone in on.

Once discovering the optimal number of clusters, K-means clustering could now be completed. This was mainly done through the incorporation of the kmeans() function, using the scaled dataset and with a centers (k-value) parameter of 4, corresponding to what was derived from the elbow plot. A random seed was set (123), in order to ensure reproducibility of the clustering. After the K-means clustering was finished, the cluster assignments were added to the original social data, helped by first converting the assignments into a factor. Finally, the clusters were visualized by the fviz_cluster() funciton, resulting in the two-dimensional cluster plot shown above. Although there is some overlap among produced clusters, this could be due to a greater proportion of users with general interests, who do not focus their posts on one particular interest category. Regardless of the slight overlap, the clusters appear mostly distinct from one another, serving as reliable market segments to be further analyzed in the following analytical steps.

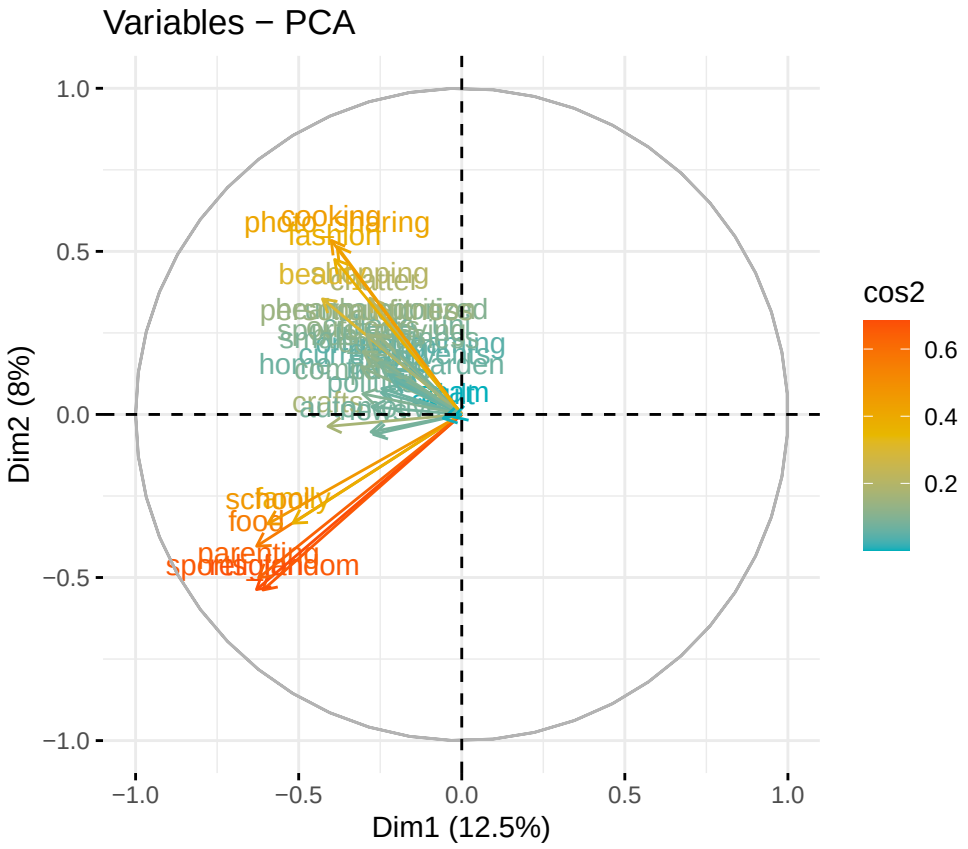
```
# Visualizing the correlation matrix with identified clusters
corrplot(corr_matrix, method = "circle", order = "hclust",
         addrect = 4, rect.col = "red")
```



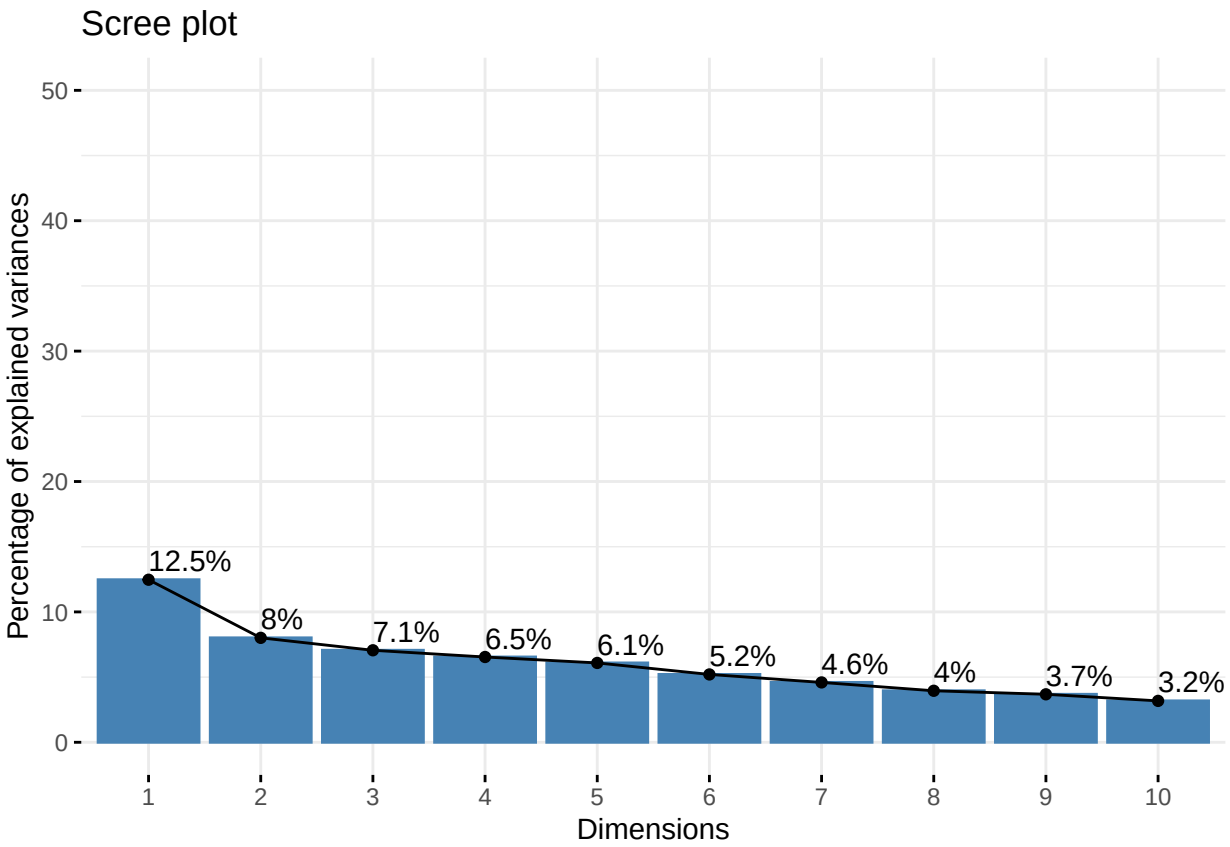
Once done with the segmentation analysis, I plotted another correlation matrix, this time with the newly identified clusters included. These clusters are represented by the large red squares in the above matrix. Again, this was done through the corrplot() function, with an addrect parameter of 4 to visualize the 4 determined clusters. From the correlation information, the contents and themes of each cluster (market segment) start to emerge. The first cluster appears to group the interest categories of school, parenting, religion, and sports fandom together, among other related interests. Although they may appear dissimilar initially, there is one commonality which they all share: the family. The four aforementioned interests represent some traditional familial values, especially in the US: education, child care and development, faith, and sports. As such, a market segment that stands out relates to family-oriented activities, which NutrientH20 can leverage in their online advertising. Another cluster that reveals itself from the matrix ties business, current events, politics, and travel together, among other interests. These are topics which working professionals and those in the business workforce tend to gravitate towards. For instance, it is important for business people to understand the current state of the markets and the destinations they may travel to for very important business meetings. Due to this, posting advertising with these themes specifically targeting working business people is likely to lead to better engagement for NutrientH20's online ad campaigns. A third cluster that is evidenced by the correlation matrix groups the categories of cooking, fashion, beauty, and health_nutrition all together, among other interest categories. Traditionally, these areas are dominated by women, who are likely to gravitate towards posts regarding new recipes, the newest lip gloss, or the latest fashion trends. While health is more of a universal interest among people, there is still a large section of women who would want to be mindful of their nutrition and physical appearance, perhaps explaining the clustering of these interests with cooking, fashion, and beauty.NutrientH20 can target this demographic in their advertising, by incorporate techniques leveraging an overarching theme of self-care, self-love, and well-being. The last cluster simply groups the spam and adult interest groups together, which really just encompasses the spam and porn "bots" that were not filtered out from the original dataset. As an alternative, NutrientH20 can focus on targeting users with generally balanced interests, who may not possess a dominant engagement level in one or two particular interest groups. That way, they can account for general social

media users who may not spent a huge section of time on social media platforms.

```
# Performing PCA
pca_result <- prcomp(social_data_scaled, center = TRUE, scale. = TRUE)
fviz_pca_var(pca_result, col.var = "cos2", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"))
```




```
# Explained variance
fviz_eig(pca_result, addlabels = TRUE, ylim = c(0, 50))
```



The last action I took in analyzing the social marketing data was to perform dimensionality reduction on the scaled dataset. The goal of this is to potentially find key factors that might offer an explanation for any variance which occurs. The first thing I did was perform principal component analysis (PCA) on the scaled data through utilizing the prcomp() function, storing the result in a variable called pca_result. After that, a PCA biplot was created using the results from principal component analysis, illustrating the variying contributions of different interest categories regarding the first two PCA components. From the biplot, there are several determinations which can be made. First off, there are a few interest categories which seem to contribute very strongly to the principal components, as indicated by their higher cosine-squared values. Among these categories are sports_fandom, parenting, and food, which are visibly projected further from the center of the circle and are red-orange in color. Several other interests seem to possess a moderate contribution to PC1 and PC2, as suggested by their yellowish color. These categories have cosine-squared values of around 0.4. Categories included in this distinction include cooking, fashion, and photo_sharing, which are a all slightly closer to the center of the circle when compared to the strong principal component contributors. Lastly, there is perhaps the greatest frequency of interests with weaker contributions to the principal components, as identified by the green-blue color these categories maintain. These interests all have cosine-squared values of less than 0.2, and are the closest to the center of the circle. Interests which fall under this distinction include politics, tv_film, and college_uni. Overall, it seems like categories such as sports_fandom, parenting, and food are potentially key components as it comes to understanding social media users, and might be interest categories worth investigating for NutrientH20. That being said, more principal components may be required to account for the full variance in the dataset.

Alongside the PCA biplot, I created a scree plot using the fviz_eig() function, in order to assess the portion of variance each principal component explains. The number of dimensions (principal componets) is on the x-axis, and the percent of variance explained is on the y-axis. As seen in the scree plot, 12.5% of the variance in the data can be explained by PC1, and about 8% of the variance in the data can be explained by PC2. With regards to the other principal components, the percentage of variance explained appears to decrease in a more gradual fashion, suggesting that the following PCs possess a lesser contribution to the

data's general variance. As such, it appears that the optimal dimensions is actually 2, as after that, each additional principal component becomes less impactful when concerning variance explanation. Therefore, it is critical for NutrientH20 to focus on the first two principal components when planning out their social media marketing strategy.

To summarize, there are four interesting market segments which appear to stand out in NutrientH20's social-media audience, as defined by marketing segmentation via K-means clustering. The first segement relates to family-oriented individuals, with interests including school, parenting, religion, and sports fandom. The second segment encompasses activities and interests possessed by working business people, containing categories like business, current events, politics, and travel. The third segment embodies interests that are traditionally feminine in nature, such as cooking, fashion, beauty, and health_nutrition. The fourth segment is tailored toward general social media users who possess balanced, non-dominant interests, or who may not be frequent posters on social media. Overall, it would be wise for NutrientH20 to tailor their online advertisement to these particular market segments, constantly maintaining their focus on what makes each segment unique. For instance, they can base their themes around school and parenting for the first segment, business and travel for the second segment, and fashion and beauty for the third segment. For the fourth segement, NutreintH20 should take a balanced approach in terms of their online advertising, honing in on more general interests and categories which embody the general social media user space. After all, they should aim to reach as many types of users as possible, not just those with dominating interests.

Name: Alexander Schmelzeis
Assignment: Exercise 7 - The Reuters corpus
Date: August 15th, 2024

The responses to this exercise are located in the Jupyter notebook: "The Reuters Corpus.ipynb" attached to the GitHub repository.

Name: Alexander Schmelzeis
Assignment: Exercise 8 - Association Rule Mining
Date: August 16th, 2024

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## v readr     2.1.5
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
##
## The following objects are masked from 'package:lubridate':
##
##     %--%, union
```

```
##
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
##
## The following object is masked from 'package:tidyr':
##
##     crossing
##
## The following object is masked from 'package:tibble':
##
##     as_data_frame
##
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
##
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
##
## The following object is masked from 'package:base':
##
##     union
```

```r
library(arules)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
##
## Attaching package: 'arules'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```r
library(arulesViz)

# Loading the data
file_path <- "C:\\Users\\aschm\\Downloads\\groceries.txt"
groceries_data <- readLines(file_path)

# Processing the data into a transactions list
```

```r
groceries_list <- strsplit(groceries_data, ",")

# Converting the transactions list to an object
groceries_trans <- as(groceries_list, "transactions")

summary(groceries_trans)
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.02609146
##
## most frequent items:
##       whole milk other vegetables       rolls/buns             soda
##             2513             1903             1809             1715
##           yogurt          (Other)
##             1372            34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.409   6.000  32.000
##
## includes extended item information - examples:
##             labels
## 1 abrasive cleaner
## 2 artif. sweetener
## 3   baby cosmetics
```

```r
# Defining the parameters for the apriori algorithm
min_support <- 0.005
min_confidence <- 0.2
min_lift <- 3

# Generating association rules
rules <- apriori(groceries_trans, parameter = list(supp = min_support, conf = min_confidence))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.2    0.1    1 none FALSE            TRUE       5   0.005      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
```

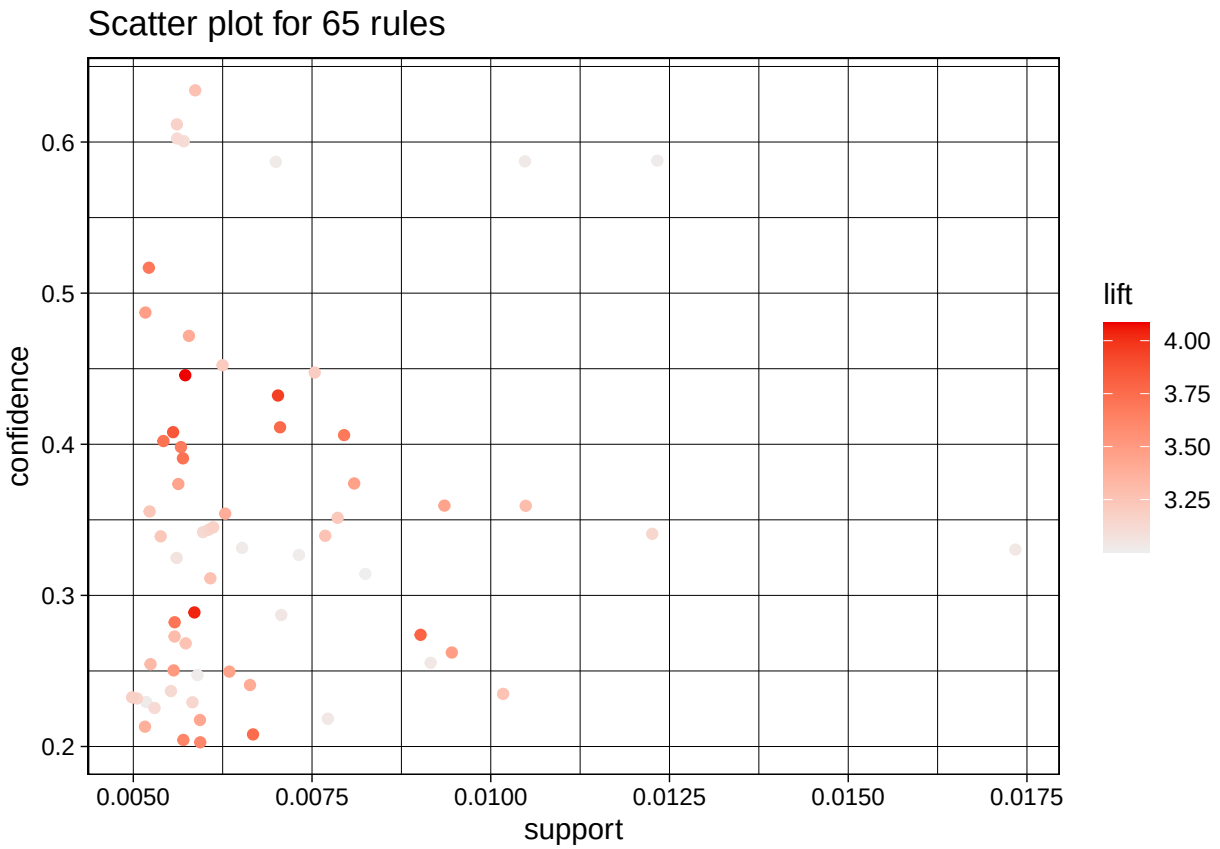
```
##
## Absolute minimum support count: 49
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [120 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [873 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
# Filtering rules by lift
rules <- subset(rules, lift > min_lift)

# Inspecting the top 10 rules by lift
inspect(head(sort(rules, by = "lift"), 10))
```

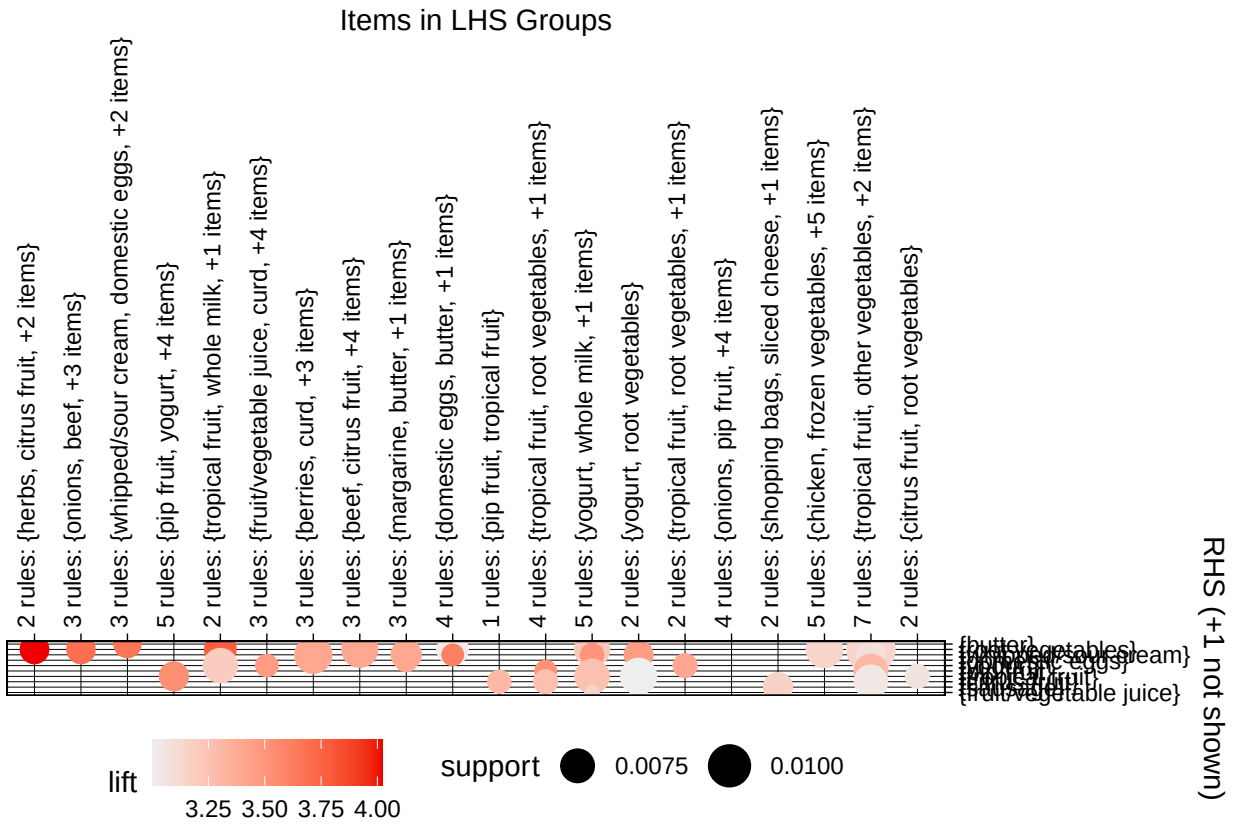
```
##      lhs                     rhs                      support confidence   coverage     lift count
## [1]  {citrus fruit,
##       other vegetables,
##       whole milk}          => {root vegetables}    0.005795628  0.4453125 0.01301474 4.085493    57
## [2]  {butter,
##       other vegetables}    => {whipped/sour cream} 0.005795628  0.2893401 0.02003050 4.036397    57
## [3]  {herbs}               => {root vegetables}    0.007015760  0.4312500 0.01626843 3.956477    69
## [4]  {citrus fruit,
##       pip fruit}           => {tropical fruit}     0.005592272  0.4044118 0.01382816 3.854060    55
## [5]  {berries}             => {whipped/sour cream} 0.009049314  0.2721713 0.03324860 3.796886    89
## [6]  {other vegetables,
##       tropical fruit,
##       whole milk}          => {root vegetables}    0.007015760  0.4107143 0.01708185 3.768074    69
## [7]  {whipped/sour cream,
##       whole milk}          => {butter}             0.006710727  0.2082019 0.03223183 3.757185    66
## [8]  {root vegetables,
##       whole milk,
##       yogurt}              => {tropical fruit}     0.005693950  0.3916084 0.01453991 3.732043    56
## [9]  {other vegetables,
##       pip fruit,
##       whole milk}          => {root vegetables}    0.005490595  0.4060150 0.01352313 3.724961    54
## [10] {citrus fruit,
##       tropical fruit}      => {pip fruit}          0.005592272  0.2806122 0.01992883 3.709437    55
```

```r
# Plotting the rules
plot(rules, method = "scatterplot", measure = c("support", "confidence"), shading = "lift")
```

```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```




```
plot(rules, method = "grouped")
```



When determining association rules for the data on grocery purchases, I first had to pre-process the data, in order to get it into the format required for association rule mining (transactional format). The first step in this process was to use the readLines() function to read all of the lines of text in the file, storing it in a variable called groceries_data. After that, the strsplit() function was utilized on the data, separating individual items on the basis of a comma, creating a list of transactions and storing it in groceries_list. Then, the as() function was used in converting the list of transactions into an object, storing it in a variable called groceries_trans. Furthermore, the summary() function was implemented on the transaction object, in order to generate some general insights on the items. As seen in the R console output, there were a total of 169 different items gathered from the object, and the summary revealed some of the most frequent items purchased. The top five items purchased based on number of purchases were determined to be whole milk (2513), other vegetables (1903), rolls/buns (1809), soda (1715), and yogurt (1372).

The next step in extracting association rules was to define the parameters for the association rules, as well as the corresponding apriori algorithm. As such, reasonable values were required for support, lift, and confidence. The support threshold represents the proportion of the transactions which contain the designated itemset. In this instance, I chose a minimum support level of 0.005, indicating that items are required to appear in at least 1 out of every 20 transactions (0.5%). Although a relatively low threshold, because there were so many transactions in the purchase data, I believe that a 0.5% transaction appearance rate was necessary, in order to account for items out of the 169 items which might not be heavily purchased. After all, I want to develop association rules for items purchased at varying frequencies, not just the most popular items. The confidence threshold represents the likelihood of the right side of the association rule occurring given that the left side of the rule happens. For the purchase data, I chose to pick a minimum confidence level of 0.2 (20%). While this is a fairly low confidence level, I did not want to gloss over any possible patterns which do not possess high confidence levels, which could provide interesting and applicable insights regarding item associations. Truthfully, I wanted a larger quantity of rules for the data, so that broader patterns can be revealed more easily. Additionally, the lift threshold represents how much more likely the items in the association rule occur together, in comparison to what would be expected if those items were

entirely independent. For the data, I chose a lift of 3, meaning that the items would be three times more likely to be purchased together than if their purchases were merely independent. A lift of 3 allows for the generation of rules where only stronger correlations between the items exist, excluding those which are not significant or likely to occur by happenstance. In other words, it ensures that the association rules for items are non simply random.

After defining the parameters, association rules were then generated on the transaction object, using the apriori() function with the determined parameters. Once these rules were generated, they were filtered based on the lift threshold set (3), meaning that only rules of lifts greater than 3 would be kept. This resulted in 65 association rules which satisfied this threshold. After filtering the rules, the top 10 rules sorted in descending order by lift were displayed, along with their associated support, confidence, coverage, and lift values. This was done by first using the sort() function to sort all of the generated rules by lift, then by utilizing the head() function to isolate only the first ten of these sorted rules. The statistics and data regarding the support, confidence, coverage, and lift of each association rule were then produced by the inspect() function. The rules were subsequently plotted on a scatter plot, with support on the x-axis and confidence on the y-axis. The lift of each of the plotted rules is represented by the color of each point on the plot, determined by a gradient. The darker red a point is, the greater lift the association rule possesses. Finally, a grouped matrix visualization was created, in order to reveal better insights relating to the generated associated rules. The left-hand side items for association rules is on the x-axis, while the right-hand side items reside on the y-axis. In the visualization, darker points represent greater lift values, while larger points indicate association rules with greater levels of support, based on the proportion of transactions they hold true in.

Several intriguing revelations can be made after examining both of the plots. When investigating the scatter plot, it is apparent that association rules with higher confidence levels typically possess lower support values. This can be observed by looking at the left side of the scatter plot, where the rules comprising of higher confidence gravitate towards a support level close to 0.005. This suggests that these rules might be applicable to fewer transactions overall, but could likely prove to be more viable and meaningful. Another pattern of note is that association rules with higher lift values are spread widely across the scatter plot. This indicates that these strong associations are not limited to lower support levels. Rather, these high-lift rules seem to exist among varying levels of confidence and support. Furthermore, the greatest number of rules seem to be packed around support values from 0.005 to 0.0075, in combination with confidence values from 0.2 to 0.4. This suggests that most of the generated association rules possess fairly low support but at least a moderate level of confidence, showing that many are reasonably reliable and worth looking at more closely. When examining the grouped matrix visualization, some interesting information is shown with regards to the properties of the generated association rules. It seems that there are a decent number of rules with both high support and lift values, as evidenced by the larger, darker circles present in the matrix. For instance, the group of rules relating to citrus fruit, pip fruit, and 3 additional items in the left-hand side and root vegetables in the right-hand side has a larger, darker red circle. This suggests a frequent association between these particular items, as well as one with a reasonable amount of significance. Also, it appears that a handful of the left-hand side groups have rules pertaining to more than one right-hand side item, indicating that these items might have the ability to predict the purchase of different kinds of products. That being said, it is unlikely that the rules' predictive effectiveness would be approximately the same for each item in the right-hand side, since the lifts of each association are noticeably different in these cases.

When interpreting the results for the generated association rules, I will be focusing on the top 5 generated rules based on lift, utilizing information from the inspect() function outputted onto the console. This is because I wanted to discover association rules where the right-hand side is most likely to be purchased, given the purchase of the items on the left-hand side of the rules. As such, the strongest association rule based on lift was citrus fruit, other vegetables, and whole milk on the left-hand side, with root vegetables on the right-hand side. This rule possessed a lift of around 4.09, meaning that the items on the left-hand side are 4.09 times more likely to be bought with root vegetables than anticipated if the items were completely independent. The support for the rule is approximately 0.0058, meaning that 0.58% of total purchases in the data consist of the items in the left-hand side and the right-hand side together. The confidence for the rule is around 44.53%, signifying that 44.53% of the shopping baskets which have citrus fruit, other vegetables, and whole milk also have root vegetables. As stated by the count, the rule applies to a total of 57 purchases. Given the results above, it is clear that the combination of items on the left-hand side strongly predicts the

corresponding purchase of root vegetables. This association makes sense to me, especially when considering the use cases for such a selection of items. Since vegetables like potatoes and carrots fall under the category of root vegetables, shoppers who purchase citrus fruit, other vegetables, and whole milk together with root vegetables might be doing so with the intent of making packed lunches. Young kids are often taught to have servings of fruits and vegetables alongside some dairy products (like milk), in an effort to develop healthy nutrition habits. Parents are cognizant of this, too, and go out of their way to ensure their children grow to their fullest potential.

The second-strongest association rule based on lift was butter and other vegetables on the left-hand side, with whipped/sour cream on the right-hand side. The lift for this rule was approximately 4.04, indicating that purchasing butter and other vegetables together increases the overall likelihood of purchasing whipped/sour cream by 4.04 times, when compared to simply random chance. The support for the rule is around 0.0058, equal to that of the first rule. As such, 0.58% of total transactions in the data consist of the items in the left-hand side and the right-hand side together. The confidence for the rule is roughly 28.93%, meaning that 28.93% of the shopping baskets which have butter and other vegetables also have whipped/sour cream. As evidenced by the count, like the first rule, the second rule applies to a total of 57 purchases. From these results, it is safe to say that there is a pretty noteworthy relationship with regards to purchasing butter, other vegetables, and whipped/sour cream. The combination of items on the left-hand side strongly predicts the corresponding purchase of whipped/sour cream. This is definitely a sensible item set, as both butter and whipped/sour cream are very frequently used as spreads or dips. For example, butter can be spread onto bread-based items such as bagels and toast, and whipped cream can be spread onto items ranging from fruits to delectable desserts (i.e, pies). The vegetables come into play when it concerns sour cream, as it is common for people to pair vegetables such as onions and celery with sour cream. Therefore, this association rule is not all too surprising to me.

The third-strongest association rule based on lift was herbs on the left-hand side and root vegetables on the right-hand side. This rule possessed a lift value of about 3.96, suggesting that if herbs are already in the shopping basket, root vegetables are roughly 3.96 times more likely to be purchased as well. The support for this rule is about 0.0070, indicating that 0.7% of total transactions in the data comprise of the items in the left-hand side and the right-hand side together. The confidence for the rule is around 43.13%, meaning that 43.13% of the shopping baskets which have herbs also have root vegetables. As seen by the count for the association rule, the third rule applies to a total of 69 purchases, more than both of the first two generated rules. Like the previous rules, there is a definite strong association that exists between purchasing herbs and root vegetables. This rule is one of the more sensible ones for me, as herbs and root vegetables have several practical uses in the world of cooking. More specifically, both items are critical when making dishes such as soups, some of the most commonplace and widespread meal types across the United States. Without herbs, the soup will likely be bland, and without vegetables, the soup probably won't be very filling. Another area in food preparation where this association comes into play relates to adding seasoning to food. For instance, meats like chicken and steak tend to be infused with herbs and spices, both in restaurants and in home cooking. These seasoned meats are regularly paired with root vegetables like potatoes to create a satisfying dish. Because of this, if a chef was to purchase herbs, it would absolutely make sense for them to also buy the root vegetable they need.

The fourth-strongest association rule based on lift was citrus fruit and pip fruit on the left-hand side and tropical fruit on the right-hand side. The lift for this rule was about 3.85, indicating that if citrus fruit and pip fruit are already in the shopping basket, tropical fruit is 3.85 times more likely to be purchased as well. The support for this rule is approximately 0.0056, suggesting that 0.56% of total transactions in the data consist of the items in the left-hand side and the right-hand side together. The confidence for the rule is roughly 40.44%, meaning that 40.44% of the shopping baskets which contain citrus fruit and pip fruit also contain tropical fruit. From the count for the association rule, the fourth rule applies to a total of 55 purchases, which is the least among the top five association rules based on lift. As was the case in the previous rules, there seems to be a clear strong association between purchasing citrus fruit and pip fruit and purchasing tropical fruit. This rule does in fact make sense to me, due to the great similarities in the recipes derived from citrus fruit and tropical fruit. For instance, when making a dish like fruit salad, it is fairly common for people to pair citrus fruits like oranges and tropical fruits like pineapples together. In some cases, people might even squeeze lemon juice over their salad. Many drinks resulting from citrus fruits

(i.e., juices) can be made in a similar process with using tropical fruits. After all, drinks like orange juice and pineapple juice are not exactly unheard of. Regardless, there is certainly enough culinary creations that can be made from these fruits to justify the willingness to buy them together.

The fifth-strongest association rule based on lift was berries on the left-hand side and whipped/sour cream on the right-hand side. The lift for this rule was around 3.80, suggesting that if berries are already in the shopping basket, whipped/sour cream is 3.80 times more likely to be purchased as well. The support for this association rule is about 0.0090, indicating that roughly 0.9% of total transactions in the data contain the items in the left-hand side and the right-hand side together. The confidence for this rule is about 27.22%, meaning that 27.22% of the shopping baskets which consist of berries also consist of whipped/sour cream. As evidenced by the count for the rule, the fifth rule applies to a total of 89 purchases, which is the most among the top five association rules based on lift. As is the common theme among the rules, there is a definite strong association between purchasing berries and purchasing whipped/sour cream. While this association might seem strange at first glance, I actually do believe that it makes some sense overall. This is because of the importance of both items when making different desserts. For instance, berries play a critical role when it comes to making cakes, such as blueberry and raspberry cakes. Often, whipped cream can be spread on top of the cakes, improving the taste and quality of the dessert. It is also fairly common to directly pair berries with whipped cream, as is the case with berries with cream. Although certainly not what initially comes to mind, there are enough dessert-related applications of berries and whipped cream to explain the generated association rule.

Name: Alexander Schmelzeis
Assignment: Exercise 9 - Image Classification with Neural Networks
Date: August 17th, 2024

The responses to this exercise are located in the Jupyter notebook: “Image Classification with Neural Networks - Alexander Schmelzeis.ipynb” attached to the GitHub repository.