|  |  |  |
| --- | --- | --- |
| A picture containing text, sign, clipart  Description automatically generated | **BOSTON**  **UNIVERSITY** | **METROPOLITAN COLLEGE** |

**AD 699 DATA MINING FOR BUSINESS ANALYTICS**

**ASSIGNMENT 5**

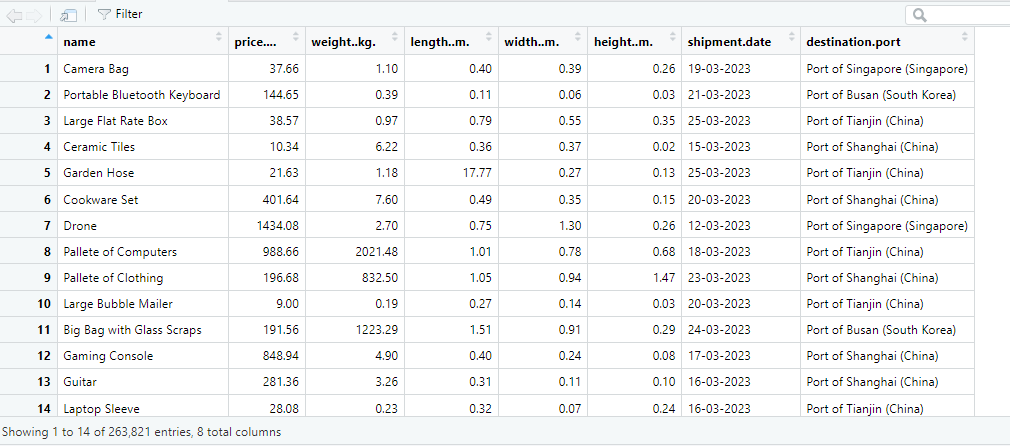
**APRIL 28, 2023**

**Aravind Hanumantha Rao**

**BU ID - U55859882**

1. Read the dataset shipping-data.csv into your R environment



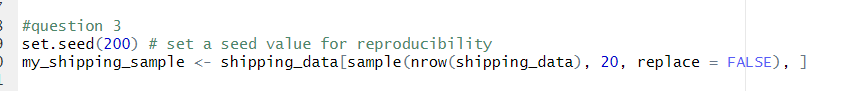


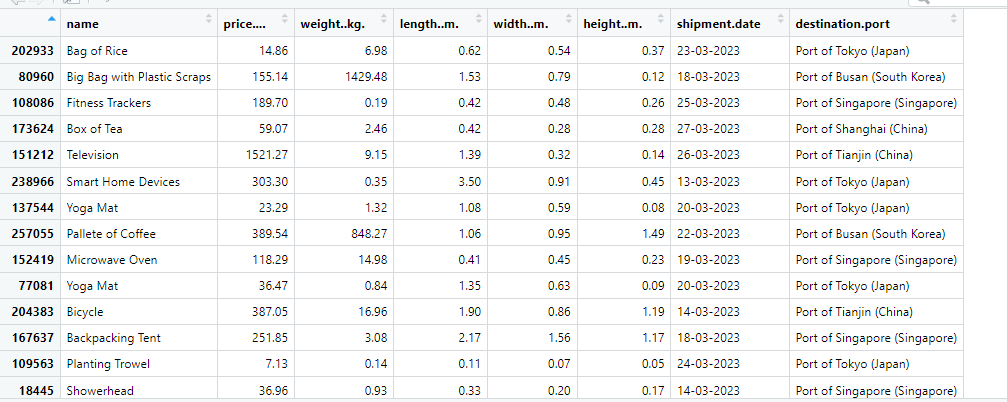
1. What are your dataset’s dimensions?

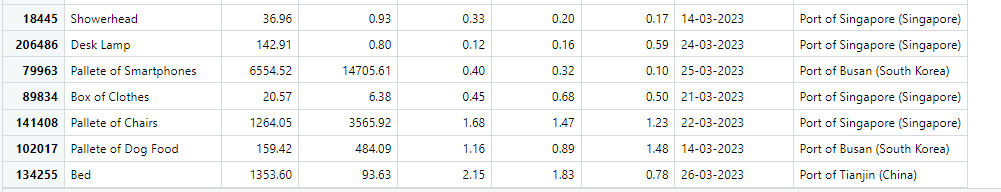


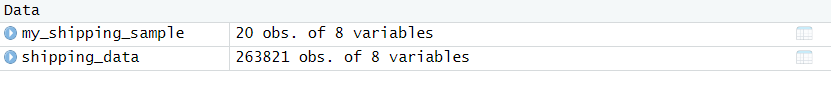
There are 263821 rows and 8 columns

3. Using any method in R for this purpose, randomly sample 20 rows from the entire group. Those are the rows that you’ll use for this clustering. You may set any seed value before sampling the data to get these 20 (I highly recommend using \*some\* seed value here, because otherwise you’ll get totally different results each time you run your script).





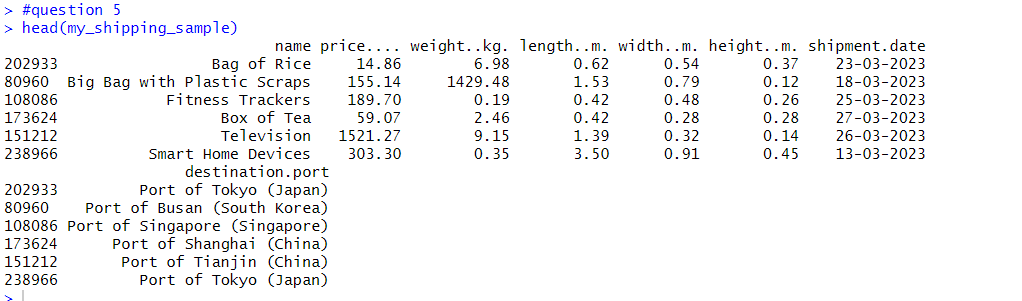




1. If any rows in your data sample contain NAs, just drop those rows entirely

There are no NA values in my dataset

5)After reading the dataset description, take a look at your data, either with the head() function or the View() function. Should your numeric variables be scaled? Why or why not? If so, then scale your data’s numeric variables?



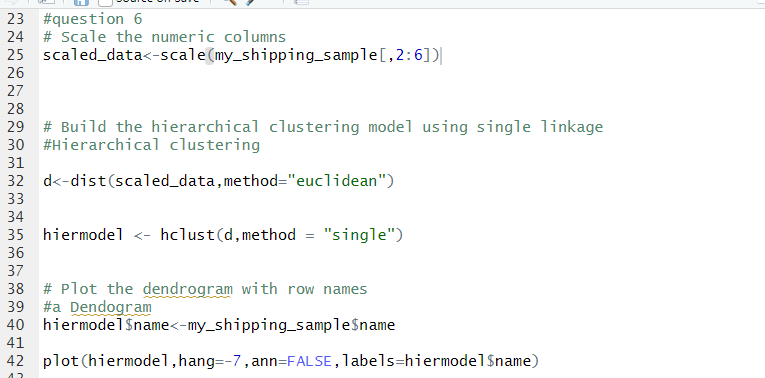
Looking at this data set, it appears that we have several numeric variables, including price, weight, length, width, and height. The units and ranges of these variables are not immediately clear, but if the variables have vastly different scales, it may be beneficial to scale them before clustering or performing any other analysis.

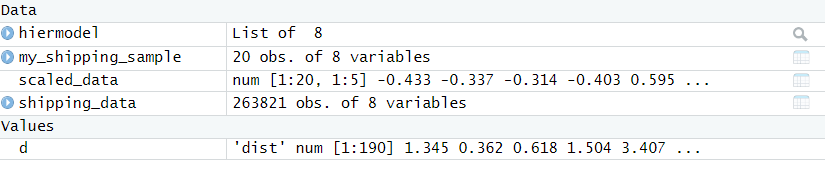
In general, it's a good practice to scale numeric variables if they have different units of measurement or if they have vastly different ranges. This is because clustering algorithms often use distance-based similarity measures, and variables with larger ranges or units may dominate the distance calculation, leading to biased results. Scaling the variables to a common range can help alleviate this issue.

Therefore, in this case, it may be a good idea to scale the numeric variables in this data set before performing any analysis.

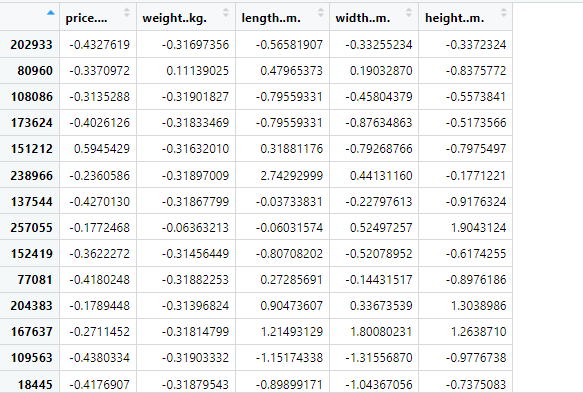
Build a hierarchical clustering model for the dataset, using any method for inter-cluster dissimilarity (if you’re not sure which one to choose you can experiment with the options from the textbook chapter). Do not use any of your non-numeric variables to build this.

a. Create and display a dendrogram for your model. Be sure that you have done this in a way that displays the ‘name’ of each item.



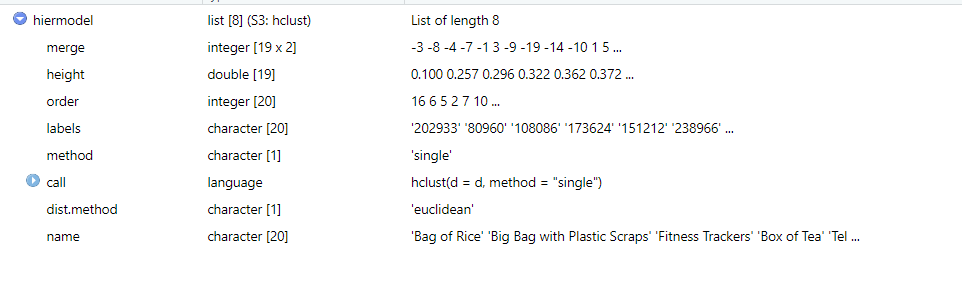


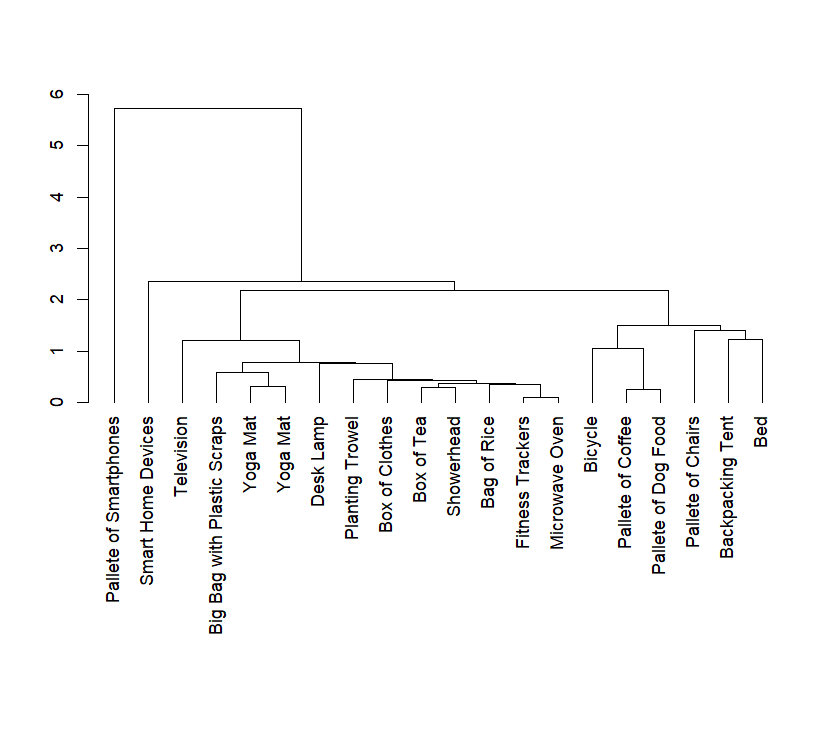
**Scaled data for the sample data**





Hier model and distance by eclidean

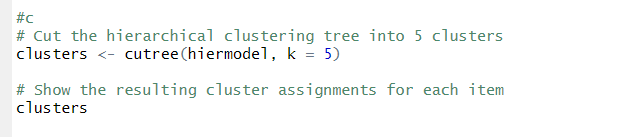


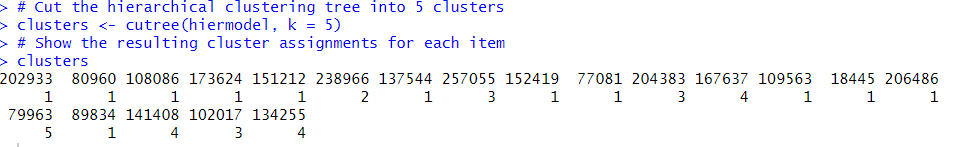


b. By looking at your dendrogram, how many clusters do you see here? (There is not a single correct answer to this question, and not all people will answer it the same way-- just describe the number of clusters that seem to be showing here)

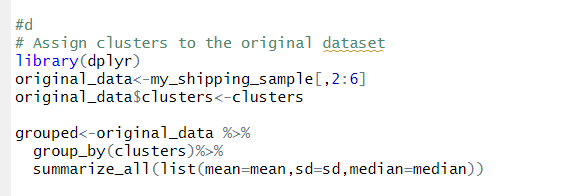
I can see 17 clusters from the dendogram

c. Use the cutree function to cut the records into clusters. Specify your desired number of clusters, and show the resulting cluster assignments for each item

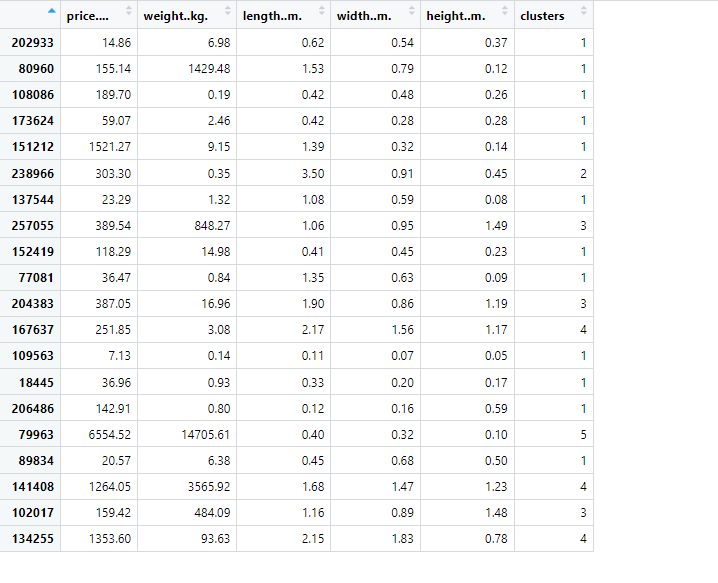




d)Attach the assigned cluster numbers back to the original dataset. Use groupby() and summarize\_all() from dplyr to generate per-cluster summary stats, and write 2-3 sentences about what you find. What stands out here? What do you notice about any unusual variables or clusters?

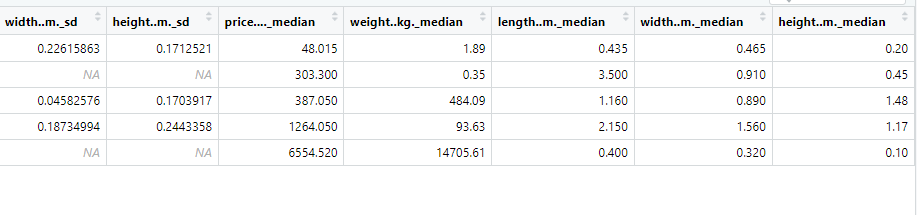


Original\_data with the clusters



Grouped data with the summary stats of median , stats and mean.

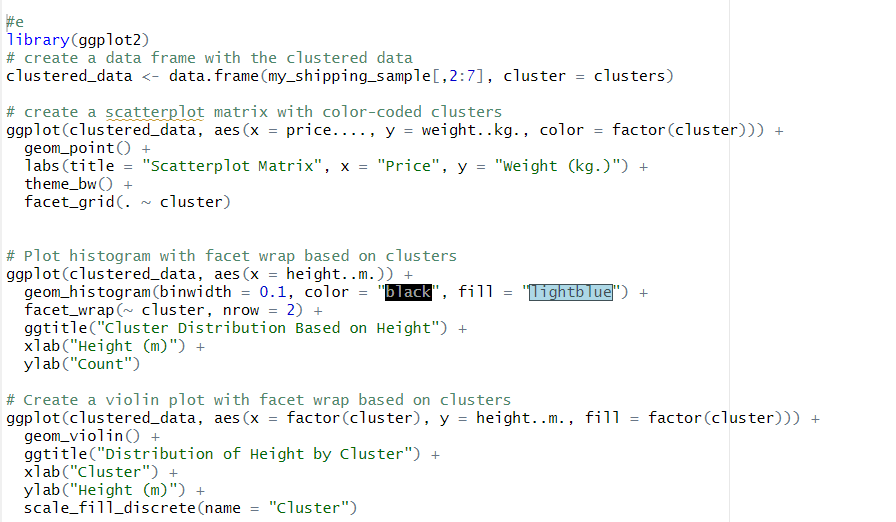
| **clusters** | | **price....\_mean** | | **weight..kg.\_mean** | **length..m.\_mean** | **width..m.\_mean** | **height..m.\_mean** | | **price....\_sd** | **weight..kg.\_sd** | | **length..m.\_sd** | | **width..m.\_sd** | | **height..m.\_sd** | **price....\_median** | **weight..kg.\_median** | | **length..m.\_median** | | **width..m.\_median** | | **height..m.\_median** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  |  |  |  |  |  |  |  |  | |  | |  | |  |  |  | |  | |
| **1** | 1 | | 193.8050 | 122.8042 | 0.6858333 | 0.4325 | 0.240000 | 422.7003 | 411.5216 | 0.5099636 | 0.22615863 | | 0.1712521 | | 48.015 | | 1.89 | 0.435 | 0.465 | | 0.20 | |
| **2** | 2 | | 303.3000 | 0.3500 | 3.5000000 | 0.9100 | 0.450000 | *NA* | *NA* | *NA* | *NA* | | *NA* | | 303.300 | | 0.35 | 3.500 | 0.910 | | 0.45 | |
| **3** | 3 | | 312.0033 | 449.7733 | 1.3733333 | 0.9000 | 1.386667 | 132.1469 | 416.7161 | 0.4588391 | 0.04582576 | | 0.1703917 | | 387.050 | | 484.09 | 1.160 | 0.890 | | 1.48 | |
| **4** | 4 | | 956.5000 | 1220.8767 | 2.0000000 | 1.6200 | 1.060000 | 611.8852 | 2031.3717 | 0.2773085 | 0.18734994 | | 0.2443358 | | 1264.050 | | 93.63 | 2.150 | 1.560 | | 1.17 | |
| **5** | 5 | | 6554.5200 | 14705.6100 | 0.4000000 | 0.3200 | 0.100000 | *NA* | *NA* | *NA* | *NA* | | *NA* | | 6554.520 | | 14705.61 | 0.400 | 0.320 | | 0.10 | |



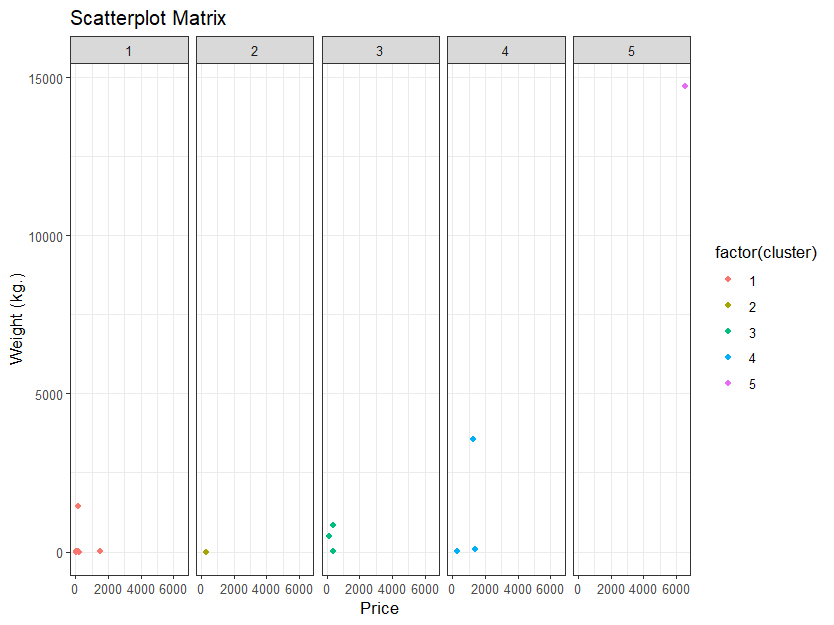
Looking at the clusters and their associated variable means, medians, and standard deviations. Cluster 1 stands out as having a much lower price mean and median compared to the other clusters, as well as lower weight, length, and width means. Cluster 4 stands out for having much higher price and weight means compared to the other clusters. Cluster 5 also appears unusual, with a high price mean but extremely low weight, length, and width means.

From the data, we can observe that there are five clusters (1-5) and some variables that have missing values, denoted by "NA". Cluster 2 seems to be unusual as it has only one observation, which may not be representative of the entire population. Additionally, there are some variables with significantly higher means and standard deviations in certain clusters, such as the "price" variable in cluster 5, which could be an outlier or skewing the data.

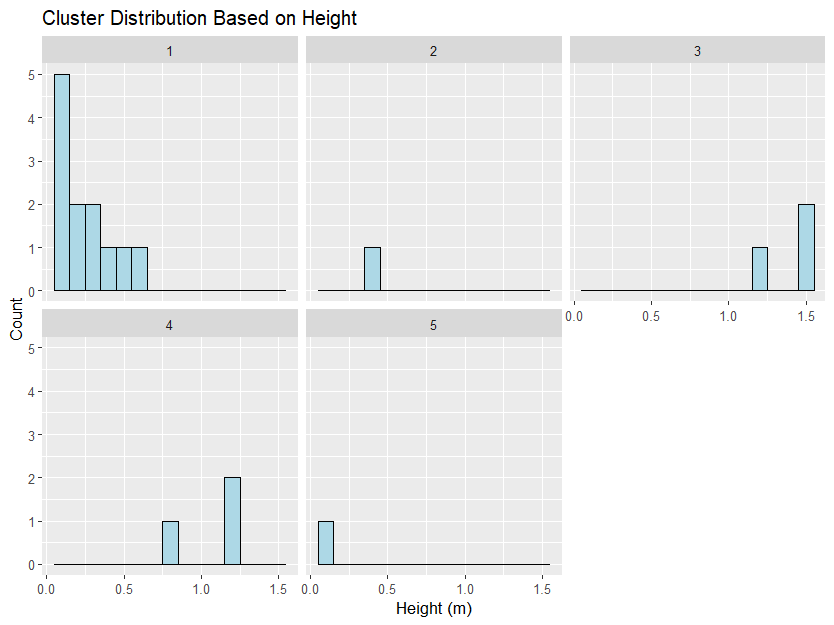
e. Make any three simple visualizations to display the results of your clustering model. Be sure that the variables depicted in your visualizations are actual variables from your dataset. Simple visualizations can include things like scatterplots, barplots, histograms, boxplots, etc.



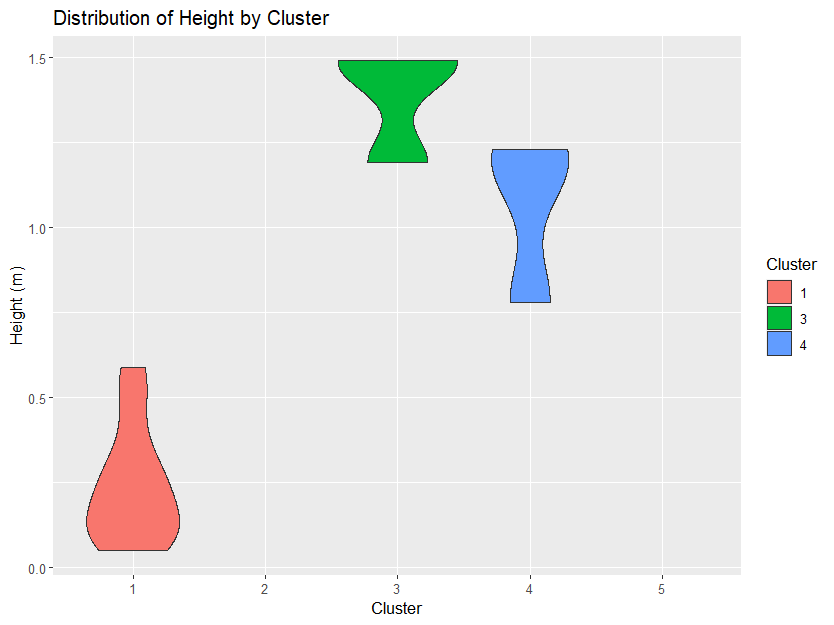
**Scatter plot**



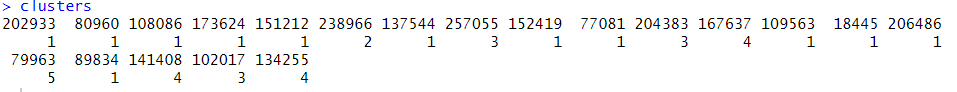
**Facet histogram**



**Violin plot**



f. Choose any item from among your sample. What cluster did it fall into? Write 2-3 sentences about the other members of its cluster (or if it’s a singleton, write a bit about why it is a singleton).



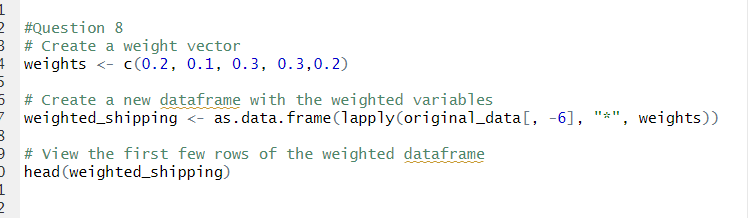


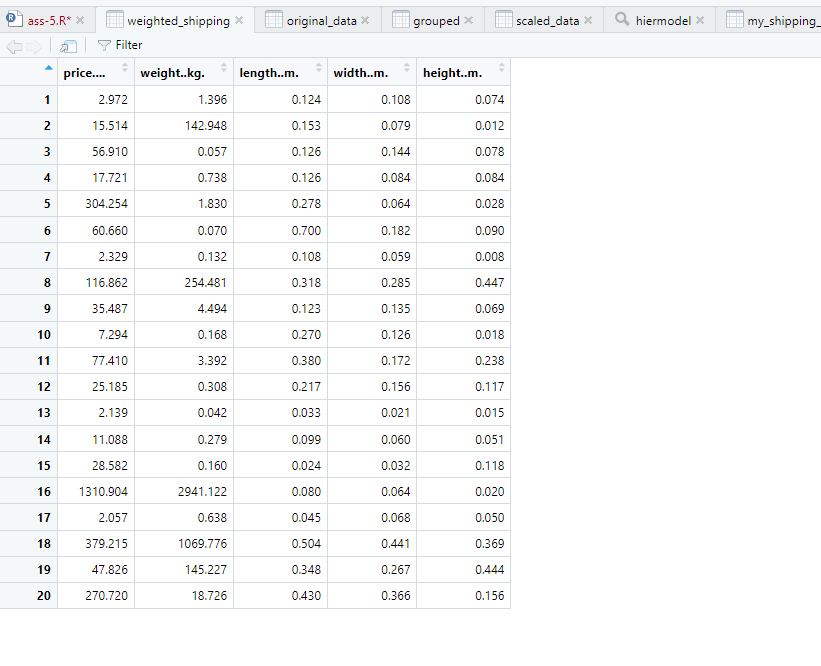
Cluster 2 is a singleton with ID 238966, and it belongs to the smart home devices category. In clustering, a singleton is a data point that forms its own cluster, as it cannot be assigned to any of the existing clusters due to being significantly different from other points in the dataset. Singletons are generally considered as outliers and can be removed to improve clustering results. However, in some cases, they can provide valuable insights into the data and should not be discarded without careful consideration.

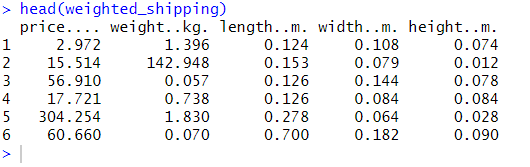
7. **In a previous step, you made the case for standardizing the variables. Now they’re all on equal footing… but why might it be problematic to view these variables with equal weight?**

Standardizing variables is a crucial step in making them comparable by putting them on the same scale. However, it's important to note that not all variables have equal importance or weight in determining the clustering structure. Neglecting variable importance can lead to incorrect or misleading interpretations of the clustering results. For instance, variables such as price, length, width, and height may have different impacts on the shipping process and should be given appropriate weights in the analysis. Therefore, it's essential to carefully consider the relevance and contribution of each variable to the research question before conducting clustering analysis.

**8. Now it’s time to fix that problem! Come up with your own weighting system for these variables, and apply it here. Be sure that you are working with a dataframe (if the data is not a dataframe, you can quickly fix that with as.data.frame(). Multiply each column by the weight that you have assigned to it. Please note that there are no rules to the weighting system– the weights do not need to add up to any particular value.**

****

****

****

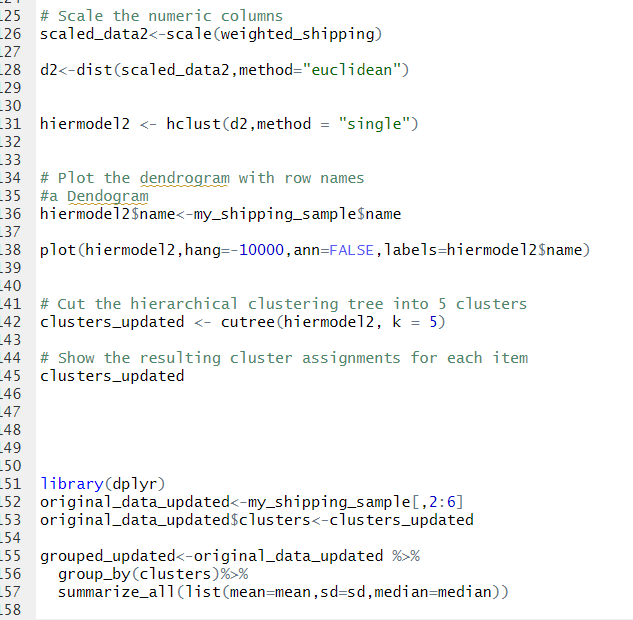
1. **Explain the weighting system in a short paragraph. There is no single \*right\* or \*wrong\* way to do this, but your answer to this question should demonstrate that you’ve taken some time to put some thought into it. One sentence per variable is enough to explain the weighting system.**

When considering the shipping cost, it's important to recognize that size-related variables such as length, width, and height may have a greater impact than weight alone. Therefore, assigning a higher weight to these variables can better capture their importance in determining the clustering structure. Additionally, the price may also play a role in determining the size of the package, so assigning a weight of 0.2 to the price variable can help account for its influence.

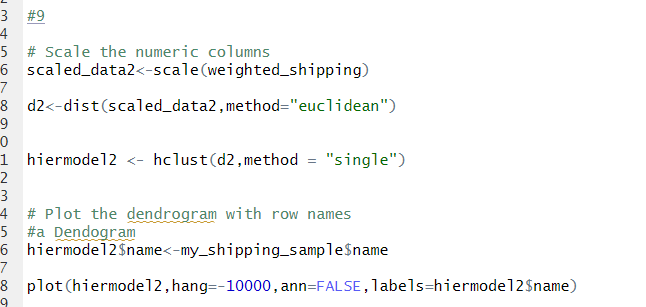
**Now, generate one more dendrogram, using your newly-rescaled set of variables (be**

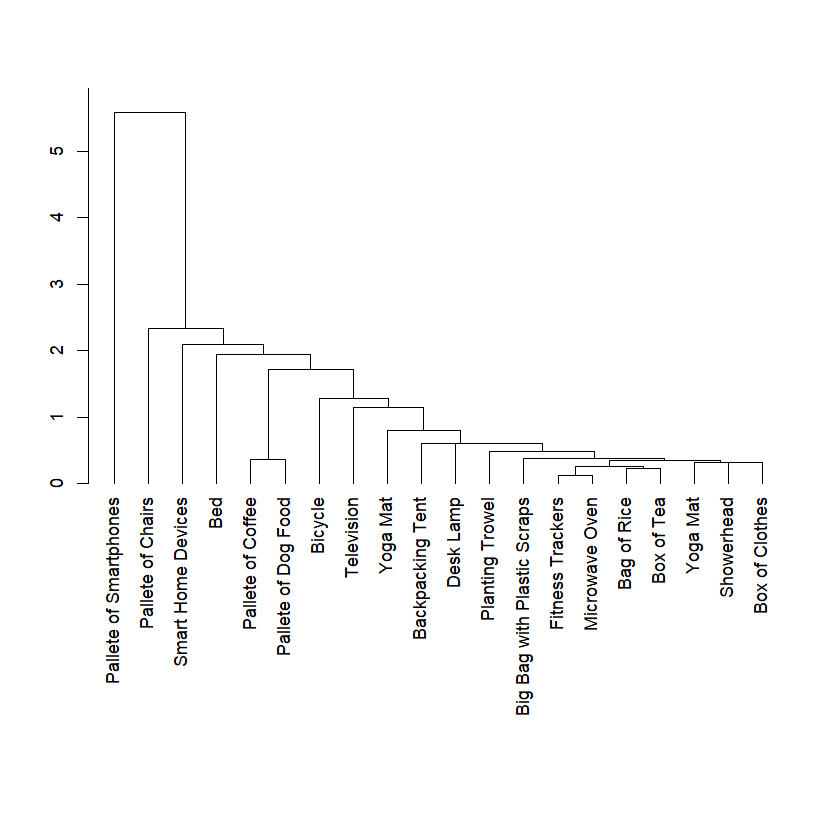
**sure that you’re not accidentally using the cluster assignments from a previous step as**

**a clustering variable here).**



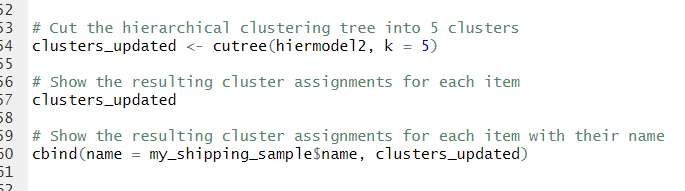
1. Once more, provide some description of what you see, and whether there are any noteworthy changes between this and the other dendrogram

****

****

The dendrogram shows that there are 18 clusters, which is one more cluster than the previous dendrogram that had 17 clusters. This suggests that the current clustering method may have identified additional patterns and variations in the data, leading to the formation of an additional cluster. It is important to carefully examine the characteristics of each cluster and assess their relevance to the research question to ensure the validity and interpretability of the clustering results.

1. Just as you did after the first hierarchical clustering, use the cutree() function to cut the records to clusters. Specify your desired number of clusters, and show the resulting cluster assignments for each state



> # Show the resulting cluster assignments for each item

> clusters\_updated

[1] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 3 1 4 1 5

> # Show the resulting cluster assignments for each item with their name

> cbind(name = my\_shipping\_sample$name, clusters\_updated)

name clusters\_updated

[1,] "Bag of Rice" "1"

[2,] "Big Bag with Plastic Scraps" "1"

[3,] "Fitness Trackers" "1"

[4,] "Box of Tea" "1"

[5,] "Television" "1"

[6,] "Smart Home Devices" "2"

[7,] "Yoga Mat" "1"

[8,] "Pallete of Coffee" "1"

[9,] "Microwave Oven" "1"

[10,] "Yoga Mat" "1"

[11,] "Bicycle" "1"

[12,] "Backpacking Tent" "1"

[13,] "Planting Trowel" "1"

[14,] "Showerhead" "1"

[15,] "Desk Lamp" "1"

[16,] "Pallete of Smartphones" "3"

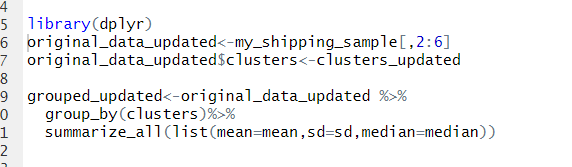
[17,] "Box of Clothes" "1"

[18,] "Pallete of Chairs" "4"

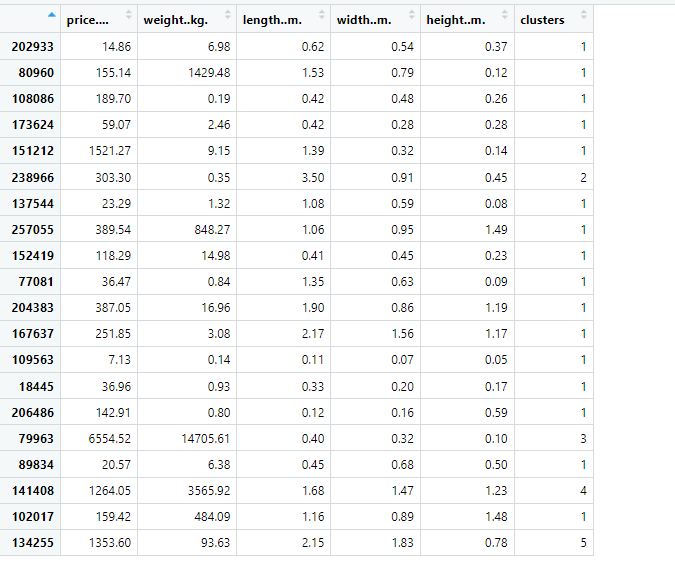
[19,] "Pallete of Dog Food" "1"

[20,] "Bed" "5"

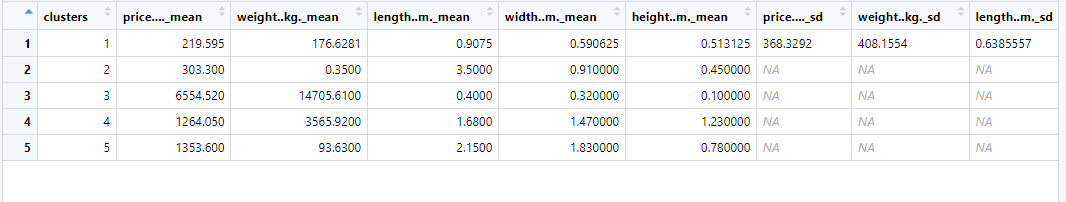
1. Attach the cluster assignments back to the original dataset. Use groupby() and summarize\_all() from dplyr to generate per-cluster summary stats, and write 2-3 sentences about what you find.

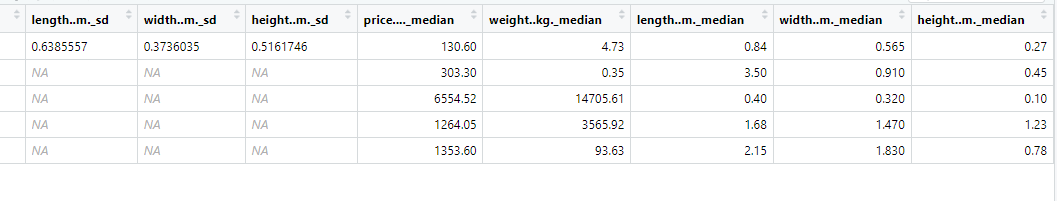


Original\_data\_updated



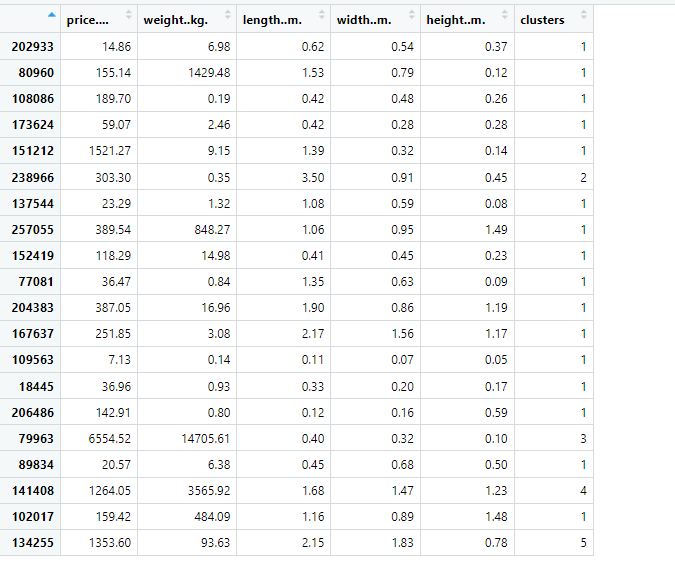
Summary\_stats



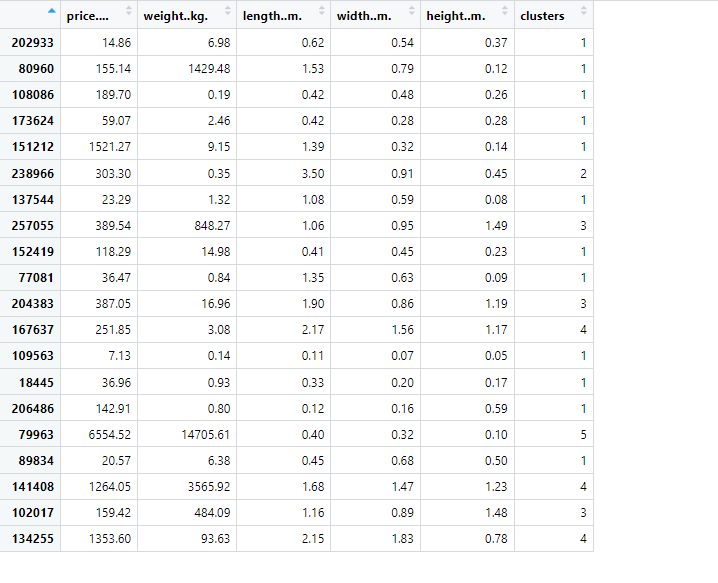


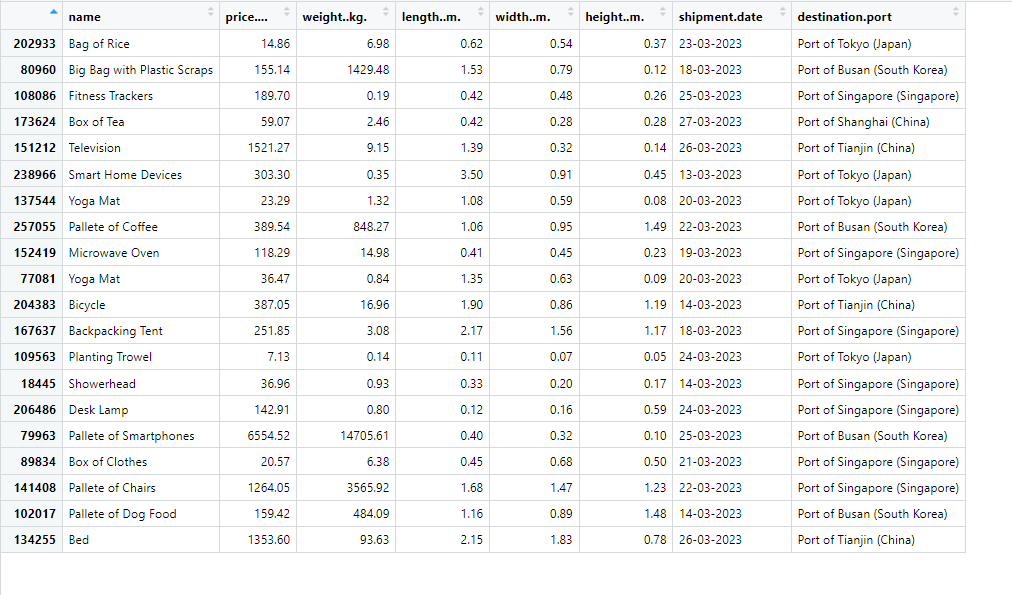
1. Let’s check back in on that item that you selected during a previous step. Where is that item now, with this new model? What else is in the same cluster? In a few sentences, talk about what changed, and why, regarding this item’s cluster assignment.

Original\_updated\_data



Original\_data





The clusters that were changed are 257055 ,167637,79963,141408,102017 and 134255 and the others remain the same .

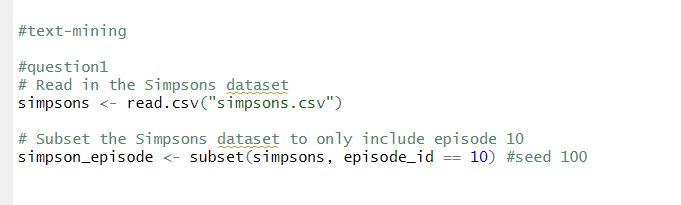
Pallete of coffee (1), backpacking trent (1), pallete of chairs(4) , pallete of dog food(1) and bed(5) . These are the new clusters associated as seen from above .

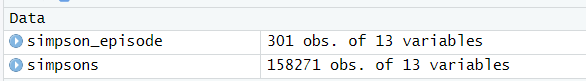
After weighting the variables, the clusters can change because the weighting changes the relative importance of each variable in determining the distance between observations. The variables that are given more weight will have a stronger influence on the distance measure used in clustering. Therefore, changing the weights can lead to a different clustering structure that better reflects the underlying relationships in the data. In the case of shipping, assigning a higher weight to the size-related variables and a lower weight to the weight variable can lead to a different clustering outcome that better captures the impact of package size on shipping costs.

**Text-mining**

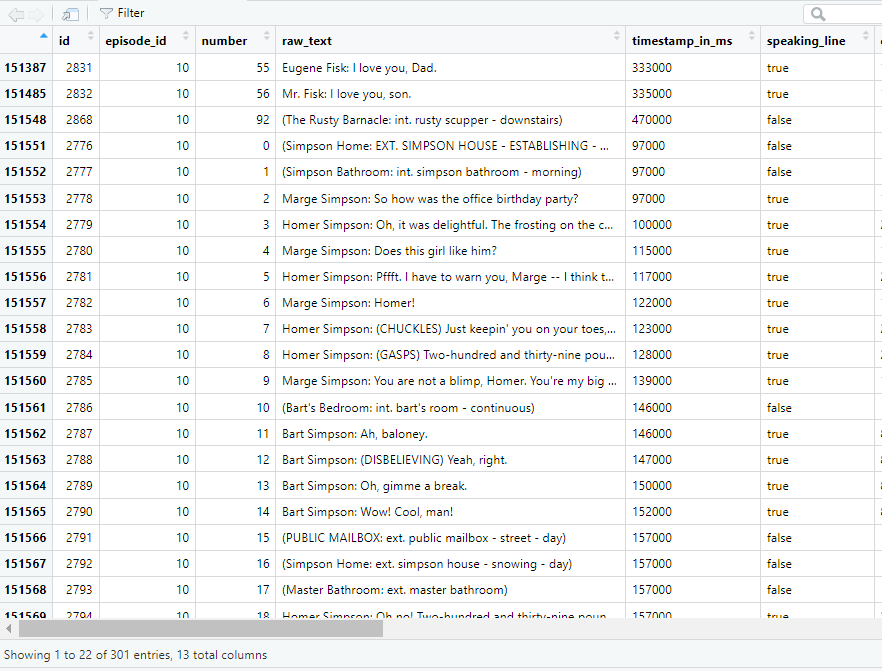
1. Load the Simpsons dataset into your R environment. Filter the dataset so that it only contains the episode that corresponds to 1/10th of your seed value from the previous two assignments (i.e. if your seed value was 20, use episode 2; if your seed value was 400, use episode 40).

My seed value is 100 , so 1/10th would be 10.

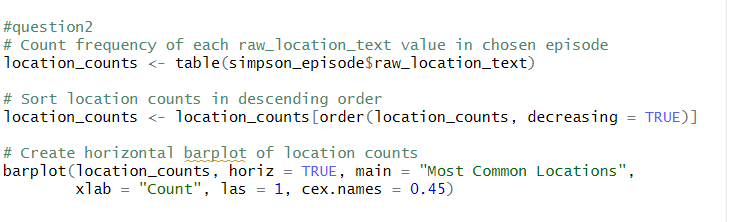


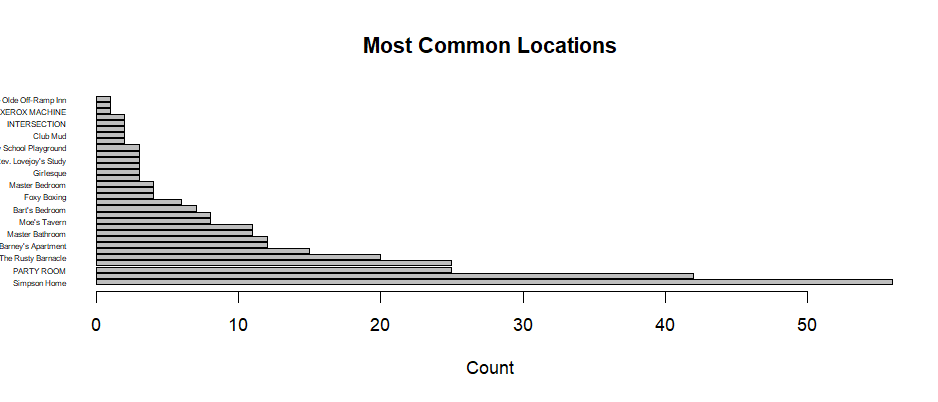


Simpson\_episode



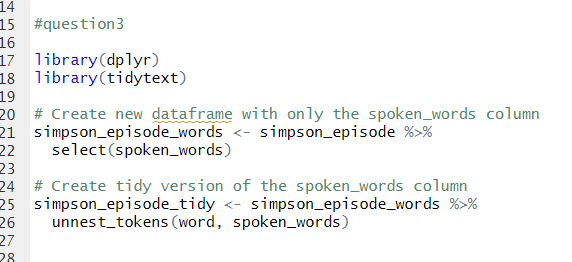
1. What were the most common locations in your episode? Create a barplot that depicts the count values for each of your raw\_location\_text values. Orient this barplot horizontally, and be sure to order your bars, either from highest to lowest or lowest to highest.



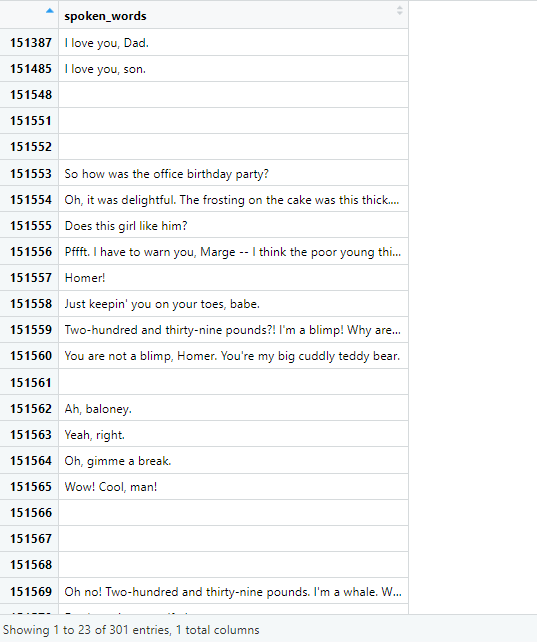




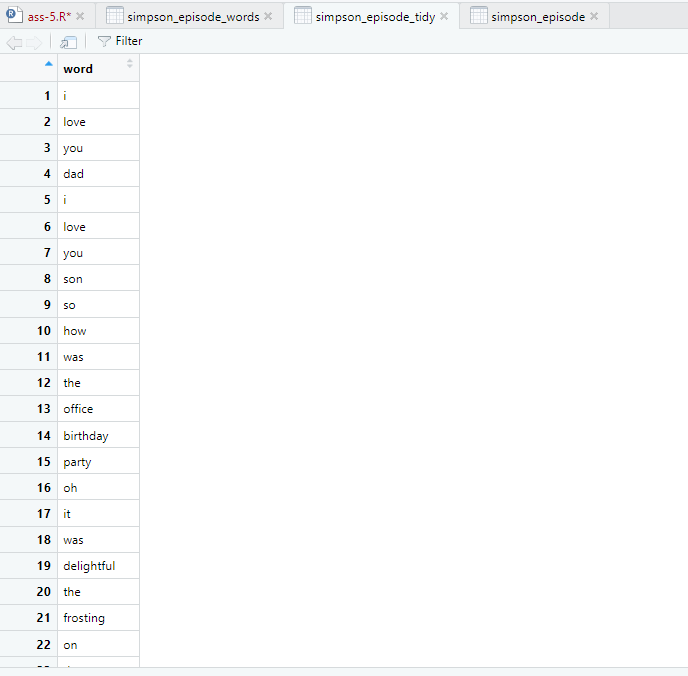
3)Using the select() function from dplyr, generate a new dataframe for your episode that only contains the spoken\_words column from your episode. a. Next, create a tidy version of your episode text. Using unnest\_tokens(word, spoken\_words) should help you to convert your text into a dataframe in which each word occupies its own row.



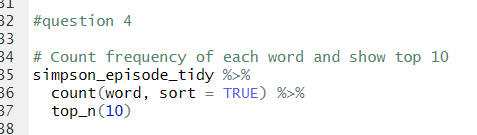
Spoken\_words



Simpson\_episode\_tidy



4)What were the 10 most frequently used words in your episode? Show the code that you used to answer this question, along with your results.



> # Count frequency of each word and show top 10

> simpson\_episode\_tidy %>%

+ count(word, sort = TRUE) %>%

+ top\_n(10)

Selecting by n

word n

1 the 91

2 i 69

3 you 69

4 a 62

5 to 53

6 my 43

7 and 41

8 of 40

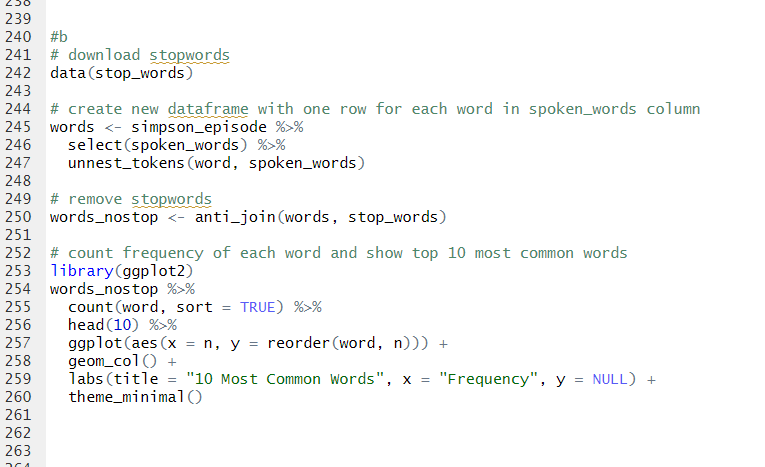
9 your 31

10 this 28

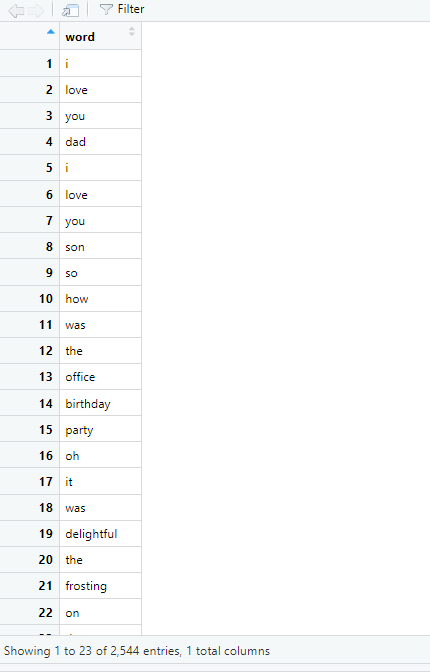
. Why is this list of very limited value for any kind of analysis?

This list of words, sorted by frequency, is limited in value for any kind of analysis because it includes very common and general words that are not specific to the episode being analyzed. Words like "the", "a", "and", and "of" are not informative in understanding the content or sentiment of the episode. They are commonly used words that appear in almost every sentence, regardless of context or sentiment. Therefore, this list does not provide any meaningful insights into the episode's language or sentiment.

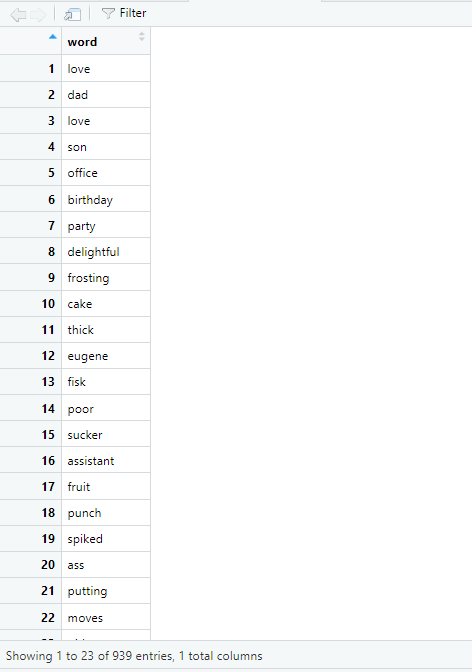
b. Now, use the anti\_join() function to remove stopwords. Show the code that you used to do this. With the stopwords removed, what are the 10 most common words in your episode? Show them here.



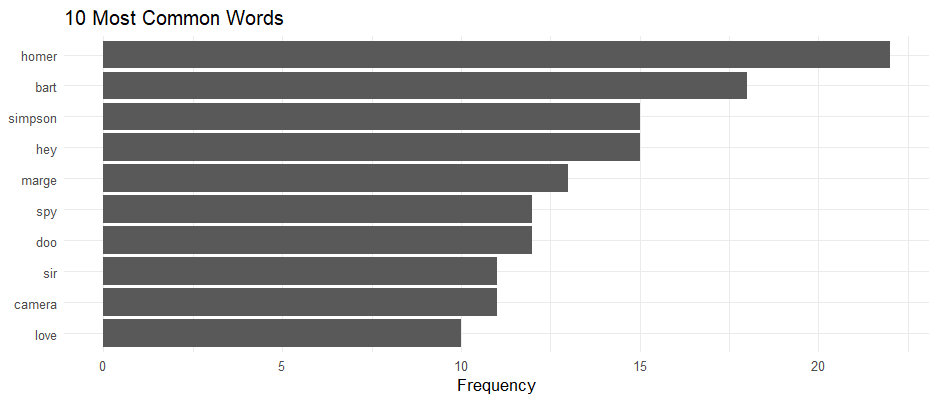
Words



Words\_nostop



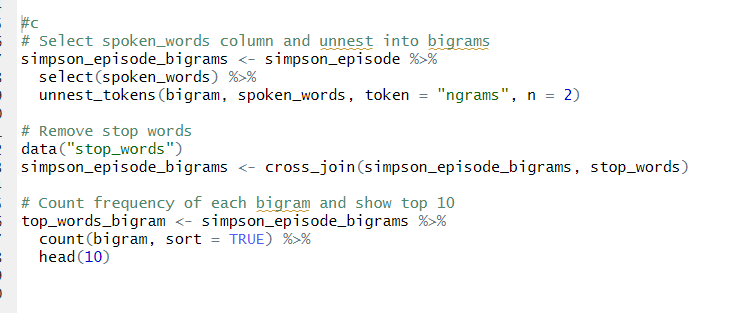
The top 10 most common words



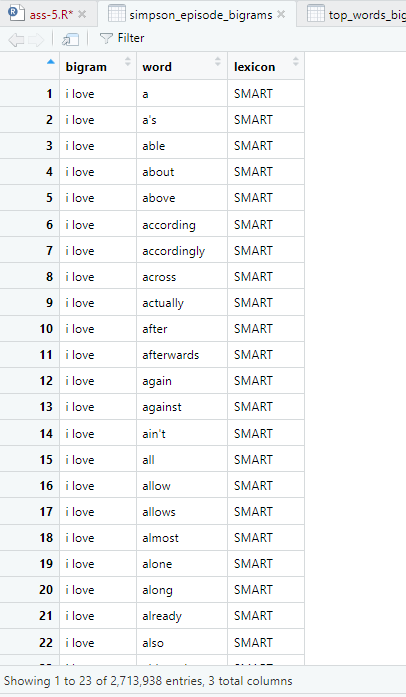
1. **Do these offer useful clues about your episode?**

Yes it does gives more information about what the episode is talking about . looks like homer is used a lot in this episode and Simpson look like the main character . it could be about a spy episode from episode 10.

c)Do this again, but instead, do it with bigrams instead of unigrams.



Simpson\_episode\_bigrams



Top 10 most common words from bigram



**How are bigrams different from unigrams?**

Unigrams refer to single words, while bigrams refer to pairs of words that occur together. For example, "happy" is a unigram, while "happy birthday" is a bigram. Analyzing bigrams can be useful in capturing more nuanced meaning and context in language, such as identifying common phrases or expressions. However, because bigrams consider pairs of words, they can also be more prone to noise and require larger amounts of data for accurate analysis.

**How might bigram analysis yield different results than unigram analysis?**

Bigram analysis and unigram analysis might yield different results because bigrams capture pairs of adjacent words, while unigrams capture individual words. Bigram analysis can provide more context than unigram analysis, as it can capture some of the meaning lost by looking at individual words in isolation. However, bigram analysis may also be more prone to noise and errors, especially if there are many rare or uncommon bigrams in the text. Additionally, the choice of n-gram size can also impact the results, with larger n-gram sizes capturing more complex relationships between words, but also potentially resulting in more noise and overfitting.

**Write 1-2 sentences that speculate about why it might be useful/interesting to**

**see this list of the most frequently-used words from your episode. What could**

**someone do with it? Use your imagination and creativity to answer?**

The list of the most frequently-used words from an episode can provide insight into the main themes and topics discussed in the episode. This information can be useful for content creators to identify popular topics and for researchers to analyze patterns and trends in language use. It can also be used for sentiment analysis by identifying the most commonly used positive or negative words.

**5. Generate a wordcloud based on your episode. You may use any wordcloud package in R, and you may set this up any way you wish to. a. What does your wordcloud show you? Describe it in a sentence or two**

#question 5

> library(wordcloud)

> data(stop\_words)

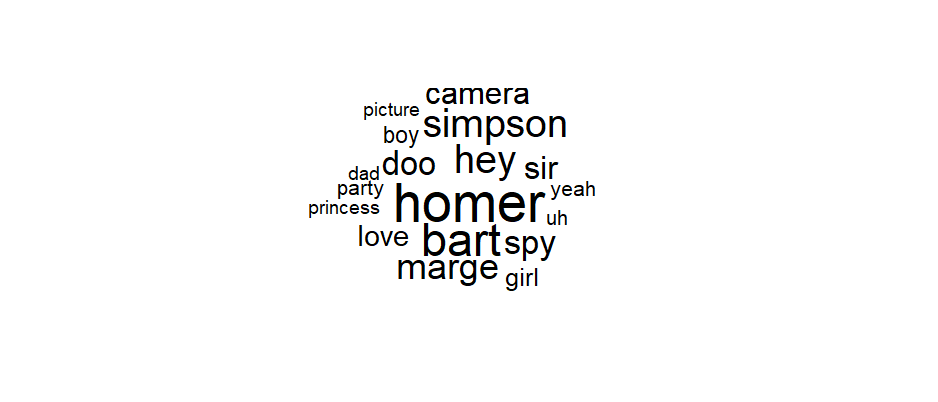
> # generate a frequency table for each word

> word\_freq <- table(words\_nostop$word)

> # create wordcloud with 20 most common words

> set.seed(100)

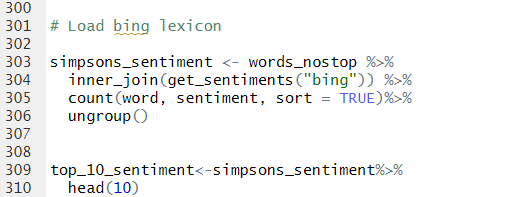
> wordcloud(names(word\_freq), freq = word\_freq, max.words = 20, random.order = FALSE, scale=c(3,0.5))

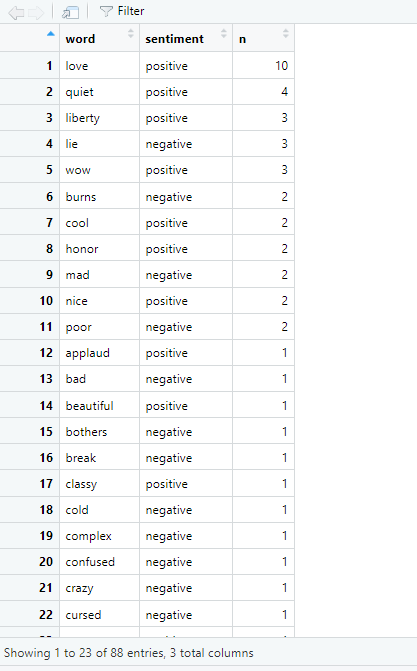


y word cloud highlights the 20 most frequently used words, with a prominent presence of words such as "Homer" and "Bart", indicating their significant usage in the analyzed episode. And the other words towards the side , which come within the 20 most frequent words.

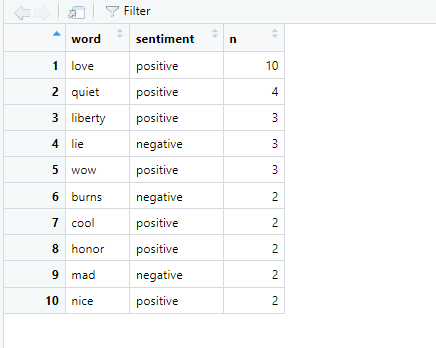
6. Next, let’s do some sentiment analysis. We will use the bing lexicon for this purpose.

a. What 10 words made the biggest sentiment contributions in your episode? Show the code that you used to find this, along with your results.





Top 10 words-



b. Of these top 10 words, how many were positive? How many were negative?

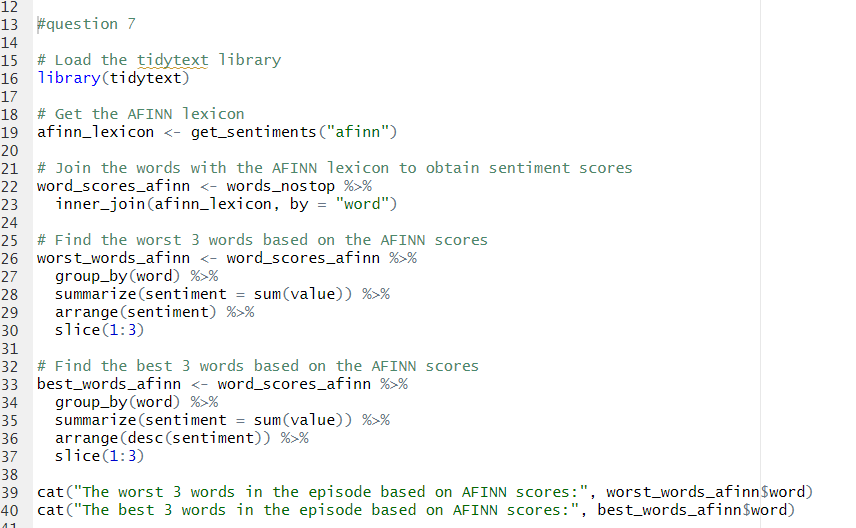
7 were positive and 3 were negative

**c. In a sentence or two, speculate about what this list suggests about your episode.**

The list suggests that the episode has an overall positive sentiment, as the majority of the words have a positive sentiment score. The frequent use of the character names "Homer" and "Bart" could indicate that they have a significant role in the episode. The presence of words like "quiet" and "liberty" could suggest themes related to peace and freedom. However, the occurrence of negative words like "lie," "burns," and "mad" may indicate that there are conflicts or challenges in the episode as well.

7. Now let’s take a look at how a different sentiment lexicon would view your episode. Bring the afinn lexicon into your environment, and join it with the text from your episode. Show the step(s) you used to do this.

a. What were the three ‘worst’ words in your episode? What were the three ‘best’ words in your episode?



Words\_scores\_affin



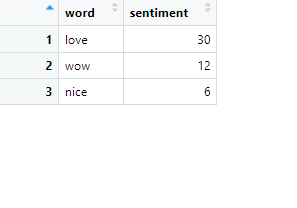
**> cat("The worst 3 words in the episode based on AFINN scores:", worst\_words\_afinn$word)**

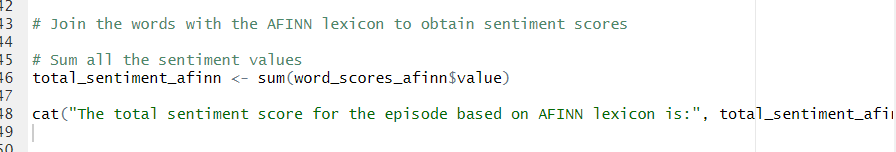
The worst 3 words in the episode based on AFINN scores: mad ass damn>



**cat("The best 3 words in the episode based on AFINN scores:", best\_words\_afinn$word)**

The best 3 words in the episode based on AFINN scores: love wow nice





b. Sum all the values for your episode. What was the total?

> total\_sentiment\_afinn <- sum(word\_scores\_afinn$value)

> cat("The total sentiment score for the episode based on AFINN lexicon is:", total\_sentiment\_afinn)

The total sentiment score for the episode based on AFINN lexicon is: 51

**What does this sum suggest about your episode? Why might this be helpful...but why might it also be incomplete or even misleading?**

The total sentiment score of 51 suggests that, on the whole, the episode has a positive sentiment. This information could be helpful in understanding the overall tone and mood of the episode. However, it is important to note that sentiment analysis is not always perfect and can be incomplete or misleading. The lexicon used may not capture the nuances and context of every word in the episode, and sentiment analysis alone cannot provide a comprehensive analysis of the episode's content and themes.