GROUP ASSIGNMENT – AMAZON REVIEW ANALYSIS

- Chaitna Mankala, Yasaswini Karanam

The dataset that we have taken is from my courses which is "iRobot Roomba 675 Robot Vacuum-Wi-Fi Connectivity, Works with Alexa, Good for Pet Hair, Carpets, Hard Floors, Self-Charging" for the utilitarian category and "New York Biology Dead Sea Mud Mask for Face and Body - Spa Quality Pore Reducer for Acne, Blackheads and Oily Skin" for the hedonic category.

Reason:

While the reviews are being scraped through amazon for the unisex sunglasses, there occurred an issue i.e. after loading 400 reviews it encountered with HTTP 404 error and it might be because Amazon might have restricted access to many review pages being scraped.

INSIGHTS

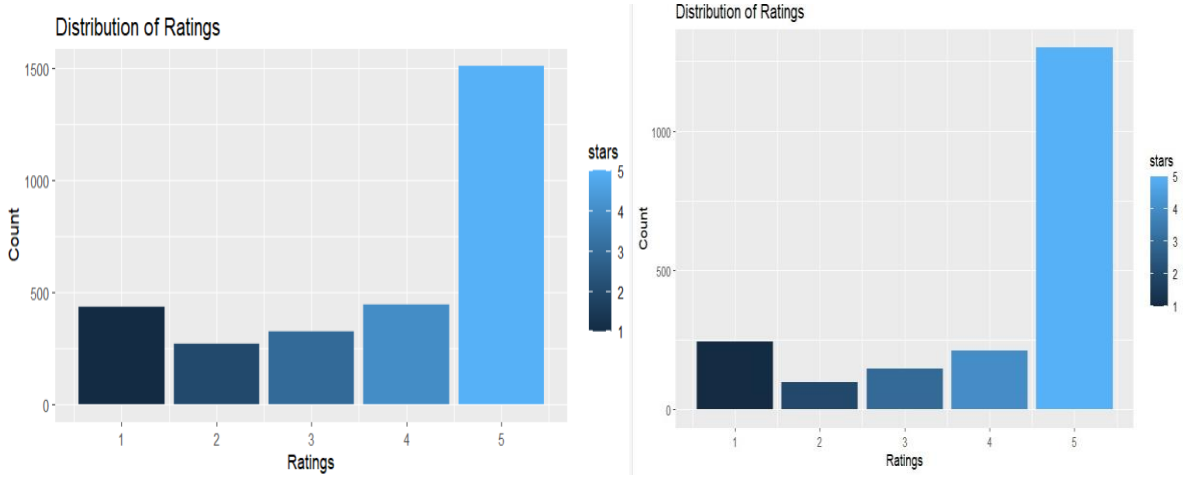
1) What are the similarities and differences between the reviews for the hedonic products and the reviews for the utilitarian products?

Word Cloud:

The above word cloud shows the most frequent words that are used in the dataset. The cloud in the left indicates the words that are frequently used in the Utilitarian category dataset and the other one is the cluster from the hedonic dataset. We can see that the word Roomba is used vastly in the iRobot Vacuum dataset, and the word hair is most frequently used in the hedonic dataset.

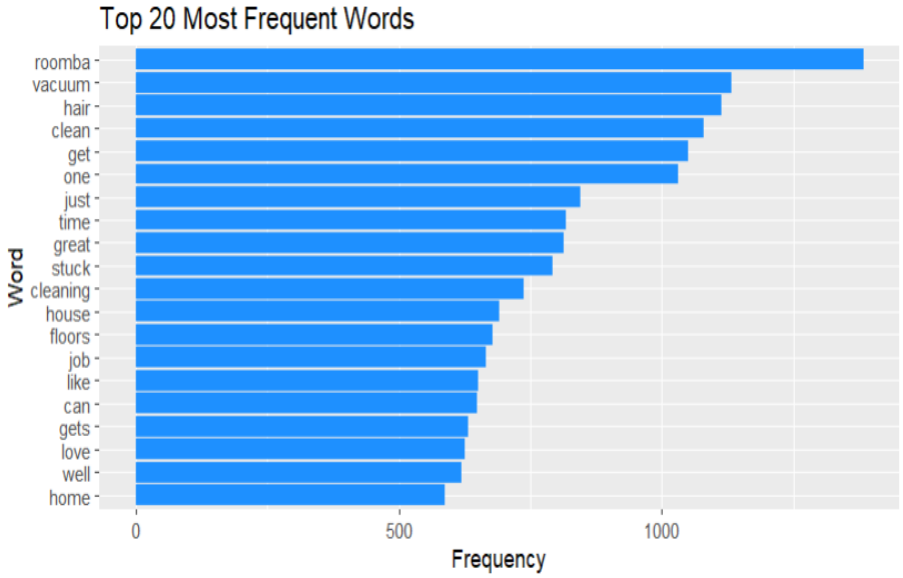
Distribution of Ratings:

The below graphs show the distributed star ratings for each category. From below, we can see that both the datasets are highly rated with five-star ratings.



Most Frequent Words:

For utilitarian product



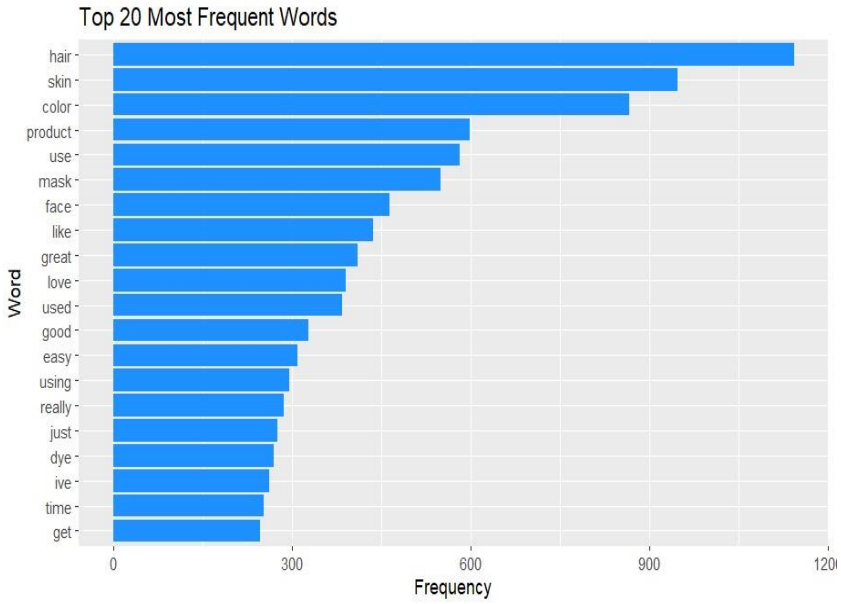
This indicates that the analysis focuses on identifying the most common words used in the dataset. These words are likely to give insight into the topics or themes most discussed in the context of Roomba vacuum cleaners.

Word Frequency: Each bar in the graph represents a specific word, and the length of the bar corresponds to how frequently that word appears in the dataset. This allows for easy visualization of which words are the most common and which are less prevalent.

Common Words: There are few words that appear frequently in the dataset. These include "Roomba," the brand name itself, which is expected to be one of the most common words. "Vacuum" is also common, as it is a core aspect of the product. Words like "hair," "clean," and "get" suggest that the dataset may include discussions about the effectiveness of the Roomba in cleaning tasks, possibly dealing with issues like pet hair. "One" could refer to specific Roomba models or instances where the word is used in the context of owning or purchasing a Roomba.

This analysis provides valuable insights into the topics and themes surrounding Roomba vacuum cleaners based on the frequency of specific words used in related texts. It can help understand customer sentiments, common problems or praises, and areas of interest or concern among users.

For Hedonic Product



This horizontal bar graph likely displays the most commonly occurring words in reviews, descriptions, or discussions about a specific type or brand of skin mask.

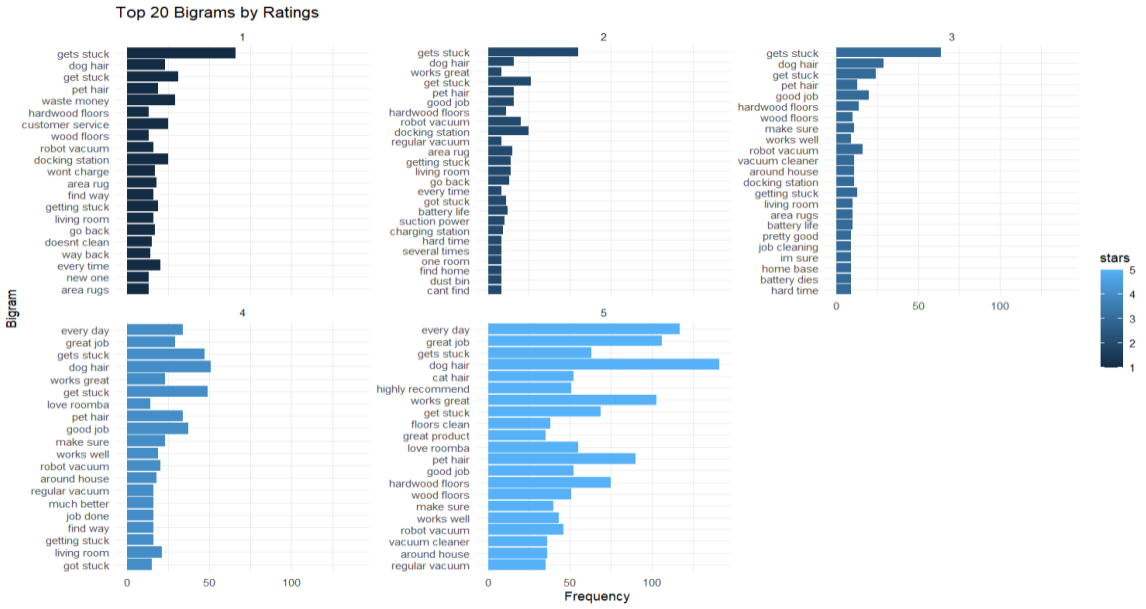
Word Frequencies: The fact that "hair" has the highest frequency might seem initially surprising in the context of a skin mask. However, it could suggest that users are discussing how the mask interacts with hair, such as if it is messy to apply or remove or if it affects facial hair. Alternatively, it is possible that "hair" is mentioned in comparison to skin concerns, like "hair removal" or "facial hair."

Other Common Words: Words like "skin," "colour," "product," and "use" are more directly related to the skin mask itself. "Skin" and "colour" might indicate discussions about how the mask affects skin tone or complexion. "Product" suggests general discussions about the mask, including its features, benefits, and perhaps comparisons to other skincare products. "Use" likely refers to discussions about how to use the mask, application techniques, or frequency of use.

Context Implication: Considering the prevalence of words related to both hair and skin, it is plausible that the dataset includes discussions about a skin mask that also has some impact or interaction with hair.

In summary, while the initial focus might be on a skin mask, the presence of words related to hair suggests that the dataset encompasses broader discussions about skincare routines, including how the mask affects different aspects of personal grooming. This nuanced understanding can provide valuable insights for product development, marketing strategies, and addressing customer concerns.

Bigrams:



This analysis offers valuable insights into the sentiments and topics discussed in reviews, grouped by star ratings, based on the frequency of specific two-word phrases (bigrams).

Utilitarian Category

Bigram Frequencies by Star Ratings: This analysis breaks down user reviews or feedback based on the star ratings they have given, ranging from 1 to 5 stars. By examining the most common two-word phrases (bigrams) associated with each rating, patterns and trends in user sentiment and experiences can be identified.

The graph likely presents a ranked list of the most frequent bigrams for each star rating category. These bigrams provide insight into the specific aspects of the product or user experience that are most mentioned in reviews across different ratings.

Patterns and Insights:

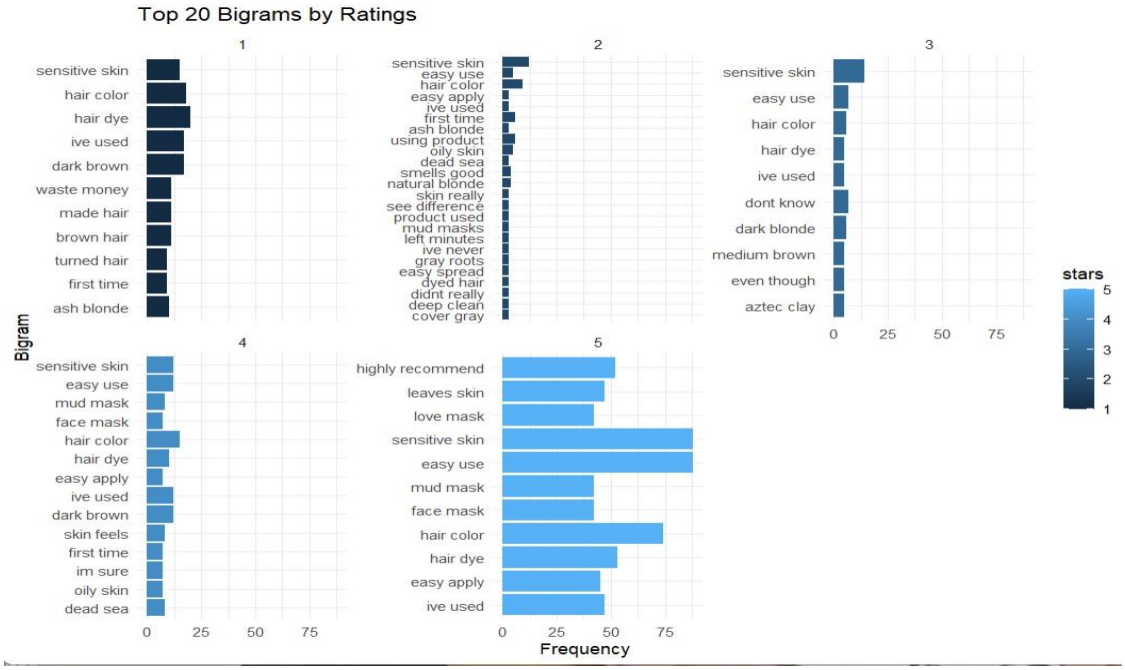
Consistent Bigrams Across Ratings: Some bigrams, such as "pet hair" and "hardwood floors," appear consistently across multiple star ratings. This suggests that certain features or issues are relevant to users across the spectrum of experiences with the product. For example, "pet hair" might indicate a common concern or benefit related to the product's performance in households with pets, while "hardwood floors" could suggest a specific type of flooring where the product is commonly used.

Rating-Specific Bigrams: Certain bigrams may be more prevalent in reviews with higher ratings. For instance, phrases like "works great" and "love pet" are indicative of positive experiences and sentiments towards the product. These bigrams highlight specific functionalities or attributes that are highly valued by satisfied customers, such as effective performance ("works great") or compatibility with pet owners ("love pet").

Insights into User Experiences and Product Features: By analysing the distribution of bigrams across different star ratings, businesses can gain insights into the aspects of the product

that contribute to positive or negative user experiences. This information can inform product development, marketing strategies, and customer support initiatives, allowing companies to address common issues, enhance desirable features, and better meet customer needs and preferences.

Overall, the analysis of bigram frequencies by star ratings offers a comprehensive understanding of user sentiments, experiences, and preferences associated with the product, enabling businesses to make data-driven decisions to improve customer satisfaction and product performance.



Hedonic Category

1-Star Ratings Insights:

Prominent Bigrams: "Sensitive skin," "hair colour," "waste money."

Common Concerns: Negative sentiments related to skin and hair care products, with specific issues such as skin sensitivity and dissatisfaction with hair colouring outcomes.

Implications: Customers in this rating category are likely expressing dissatisfaction with the performance, effects, or value of the products. Issues related to skin sensitivity and undesirable hair colouring outcomes indicate areas where the products may have failed to meet expectations or caused adverse reactions.

5-Star Ratings Insights:

Prominent Bigrams: "Highly recommend," "leaves skin," "love mask."

Positive Sentiments: Reviews express satisfaction and positive experiences with the products, with customers highly recommending them and expressing love for their effects.

Implications: Customers in this rating category are overwhelmingly satisfied with the products, praising their effectiveness and benefits. The positive sentiments indicate that the products are meeting or exceeding expectations, resulting in high levels of customer satisfaction and loyalty.

General Observations:

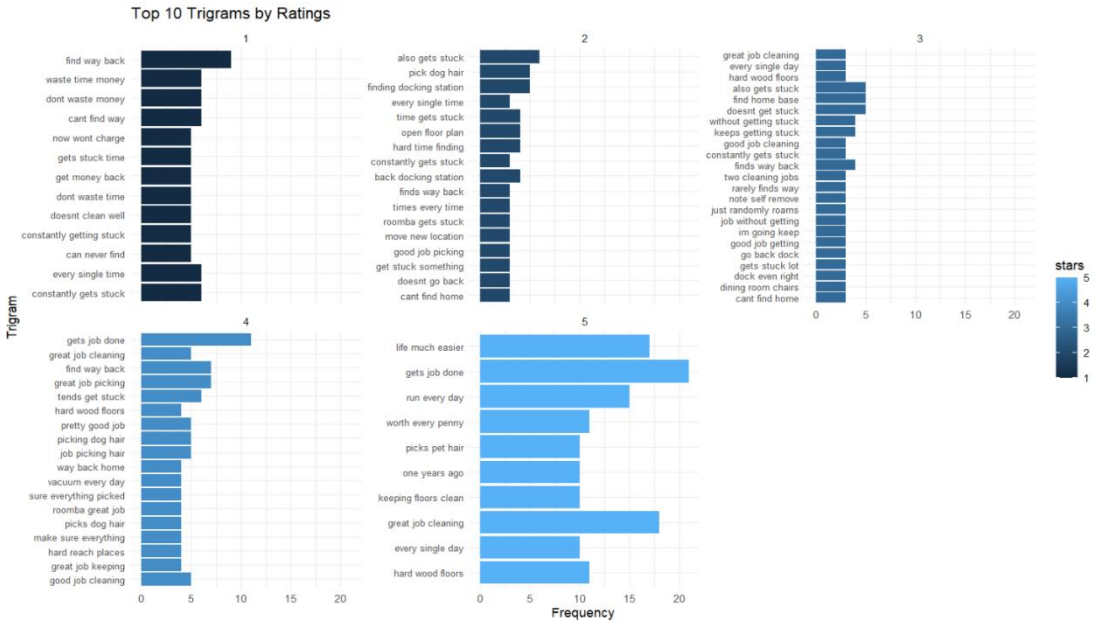
Consistent Themes: Despite differences in sentiment, bigrams related to skincare, hair colour, and masks appear frequently across both 1-star and 5-star ratings.

Divergent Sentiments: While negative sentiments dominate in 1-star ratings, positive sentiments prevail in 5-star ratings.

Areas for Improvement vs. Strengths: Insights from 1-star ratings highlight areas for improvement, such as addressing issues with skin sensitivity and hair colouring outcomes, while insights from 5-star ratings underscore the strengths of the products, such as their effectiveness and ability to leave skin feeling good.

In summary, the comparison of bigrams across different star ratings provides a nuanced understanding of customer sentiments, highlighting areas where products excel and areas where improvements may be needed. This information can guide businesses in refining their products, addressing customer concerns, and enhancing overall customer satisfaction and loyalty.

Trigrams:



For Utilitarian Category

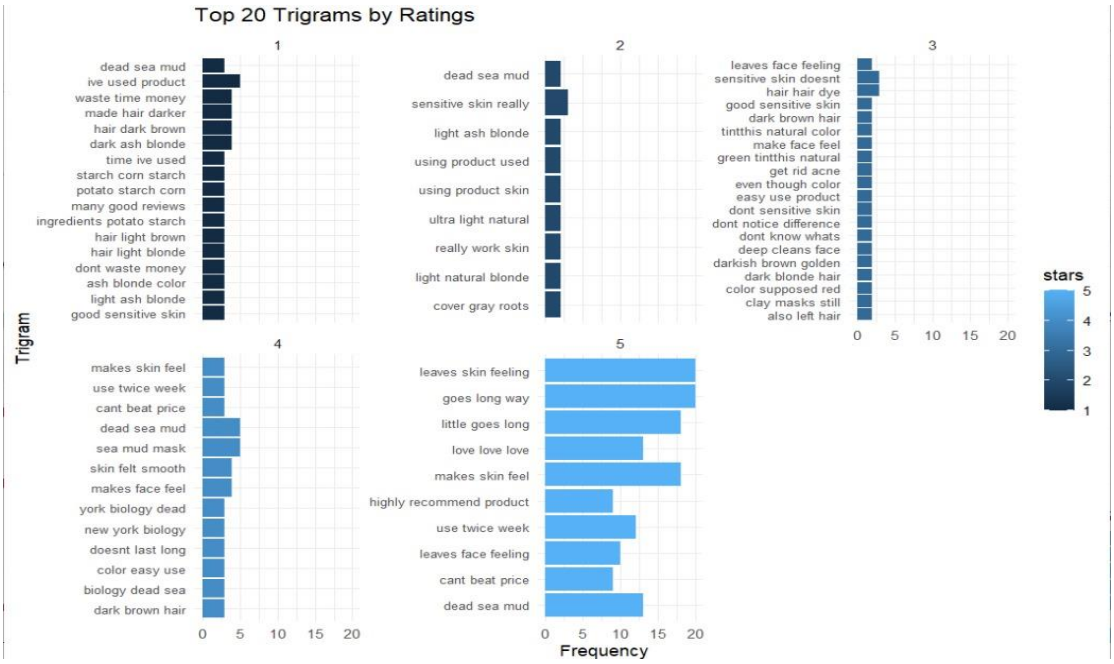
Trigrams are sequences of three consecutive words used in text analysis to identify patterns and extract key phrases

0 Stars Trigrams: Reflect dissatisfaction or issues with phrases like "can’t save money" and "never really work," indicating areas for improvement.

1 Star Trigrams: Mixed sentiments with phrases like "just gets loose" and "pretty good job," hinting at dissatisfaction despite some positive aspects.

2 Stars Trigrams: Reveal dissatisfaction with phrases like "also gets stuck" and "leaky pickup valve," highlighting specific product issues.

5 Stars Trigrams: Positive sentiments expressed through phrases like "great job cleaning" and "really well sticky," indicating high satisfaction levels and product strengths.



For Hedonic Category

The chart groups trigrams from product reviews by star ratings, allowing comparison across satisfaction levels.

Notable trigrams include "dead sea mud," "sensitive skin," and hair-related phrases like "dark brown hair" and "light ash blonde."

Specific Trigrams Insights:

"Dead sea mud" is frequently mentioned, possibly indicating a significant product feature.

Reviews discuss product effectiveness for "sensitive skin" and mention hair-related phrases, providing insights into product versatility.

Review Insights:

Some users note drawbacks like a green tint and stiffness after use.

However, the product is recommended for smooth skin and ease of use.

In summary, both the analysis highlights common phrases in reviews, mentions specific product features, and provides insights into both positive aspects and potential drawbacks.

2) Are hedonic product reviews in general different from the utilitarian product reviews in any meaningful ways or are they just similar to each other. Provide details like words used, topic models, the type of words used to describe these products, parts of speech, emotions expressed etc. Provide a clear narrative.

Average length of the text:

For Utilitarian Product

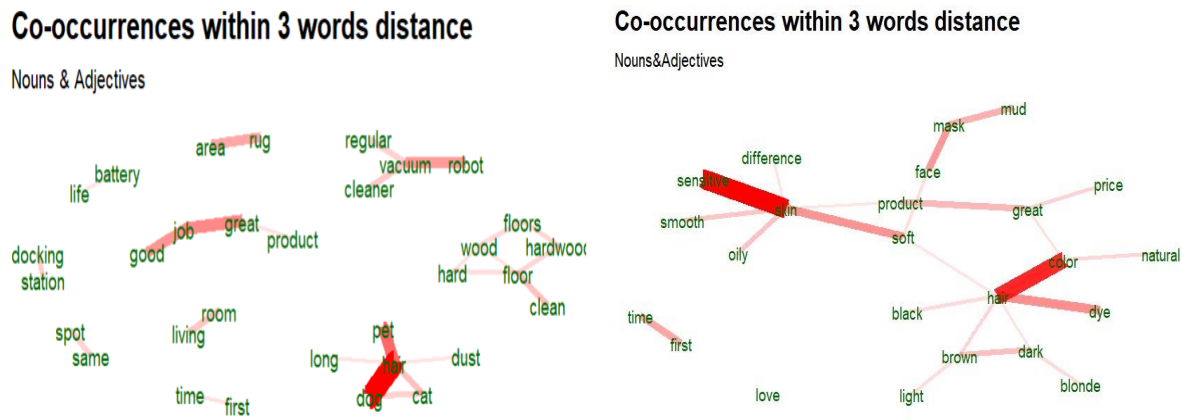
```
> print(paste("Average length of the text:", round(average_length, 2)))
[1] "Average length of the text: 73.03"
```

For Hedonic Product

```
> print(paste("Average length of the text:", round(average_length,2)))
[1] "Average length of the text: 43.63"
```

Utilitarian product reviews are lengthier on average (73.03 words) than Hedonic product reviews (43.63 words). This could mean that buyers give more specific and thorough feedback on utilitarian products. They might go over various features, performance, and usefulness in further detail. The greater average length of Utilitarian product evaluations indicates that customers are more likely to provide extensive accounts of their experiences with utilitarian products. They may describe how the product solves a specific problem or addresses a real need in their lives. However, the shorter average length of Hedonic product reviews may indicate a more concise expression of happiness or dissatisfaction with the product. Consumers may prioritise the emotional components of their experience, such as how the product makes them feel, over detailed functionality or features.

Placement of words in general:



The plot on the left for utilitarian product shows that

Word Clusters: The map shows clusters of words that frequently appear together, such as "battery life," "docking station," and "spot," which may relate to robot vacuum cleaners.

Positive Feedback: Words like “area rug,” “living room,” and “great job” suggest positive reviews about the product’s performance in specific areas.

Cleaning Context: The connection between “regular vacuum,” “robot cleaner,” and “hardwood floors” indicates discussions about different types of vacuums and the surfaces they clean.

Pet-Related Cleaning: The link between “pets,” “dogs,” “cats,” and “dust” highlights common cleaning challenges related to pets.

The plot on the right for hedonic product shows that

Contextual Associations: The visualization shows how certain words are contextually related to each other, such as “sensitive,” “smooth,” “oily,” and “natural” being connected to “mask,” “face,” and “product.”

Colour Descriptions: Words like “black,” “brown,” “light,” “dark,” and “blonde” are included, indicating that the analysis may involve descriptions of skin colours, related to our product which is face mask.

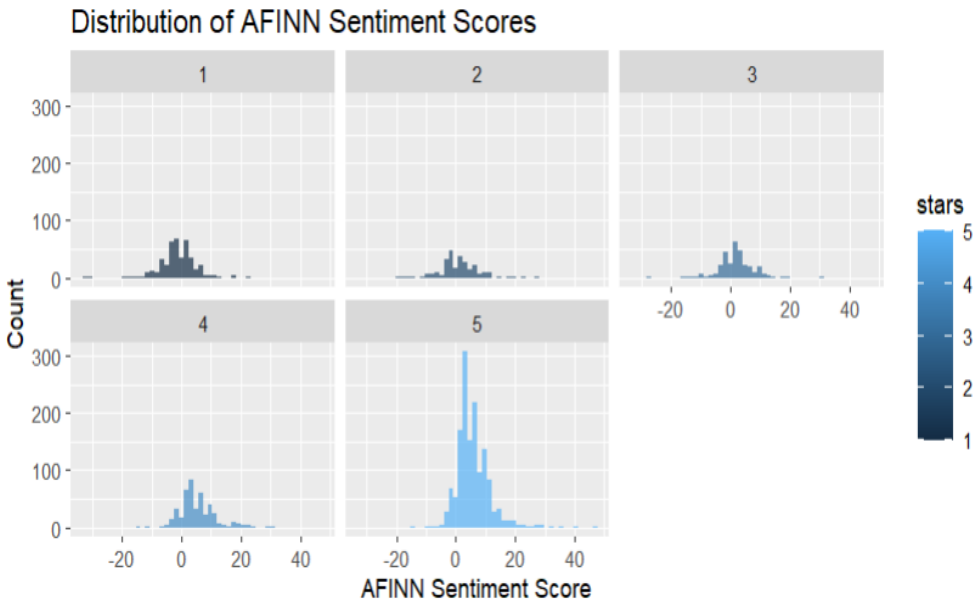
Temporal and Qualitative Aspects: The presence of words like “time,” “first,” and “great” implies that the text from which the data was extracted might discuss the timing of usage or the quality of the product.

Price Considerations: The term “price” is linked to both positive (“great”) and neutral (“colour,” “natural”) terms, suggesting a discussion about the cost-value relationship of the product.

These insights can be valuable for understanding consumer perceptions, product positioning, and marketing strategies for our product.

Sentiment Analysis:

For utilitarian product (AFINN)



The above graph shows that

1 Star: Shows a negative sentiment, with scores clustering around -10, indicating dissatisfaction.

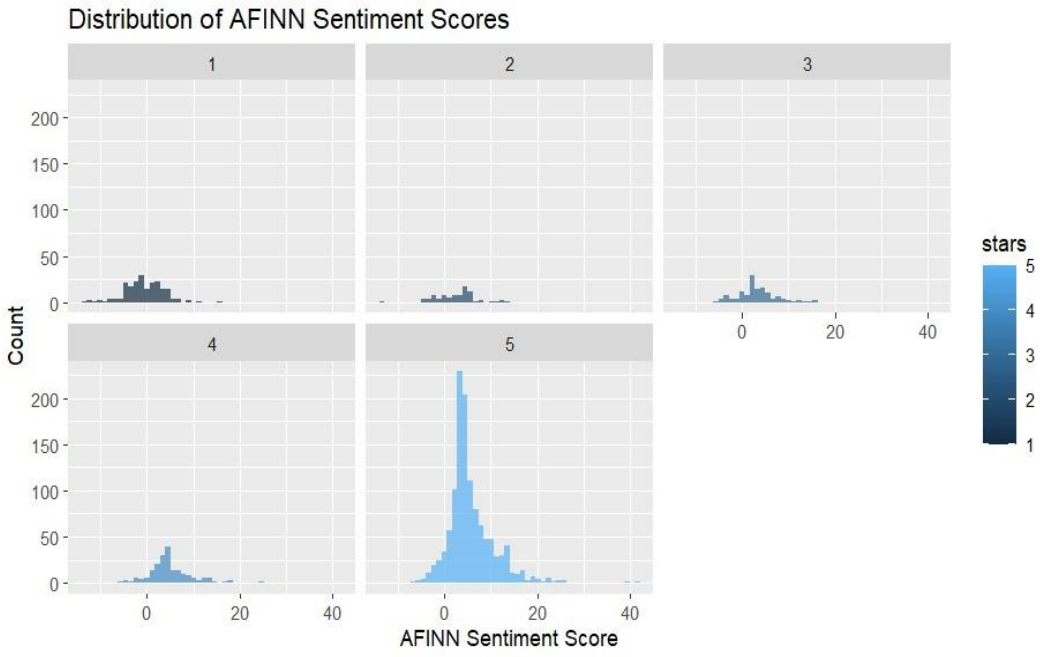
2 Stars: Also, negative but with a lower frequency of extreme sentiments compared to 1 star.

3 Stars: A neutral sentiment with a balanced distribution, suggesting mixed reviews.

4 Stars: Displays a positive sentiment, though spread across a range, indicating varied but generally good experiences.

5 Stars: Exhibits a strong positive sentiment, peaking around a score of 10, reflecting high satisfaction.

For Hedonic Product (AFINN)



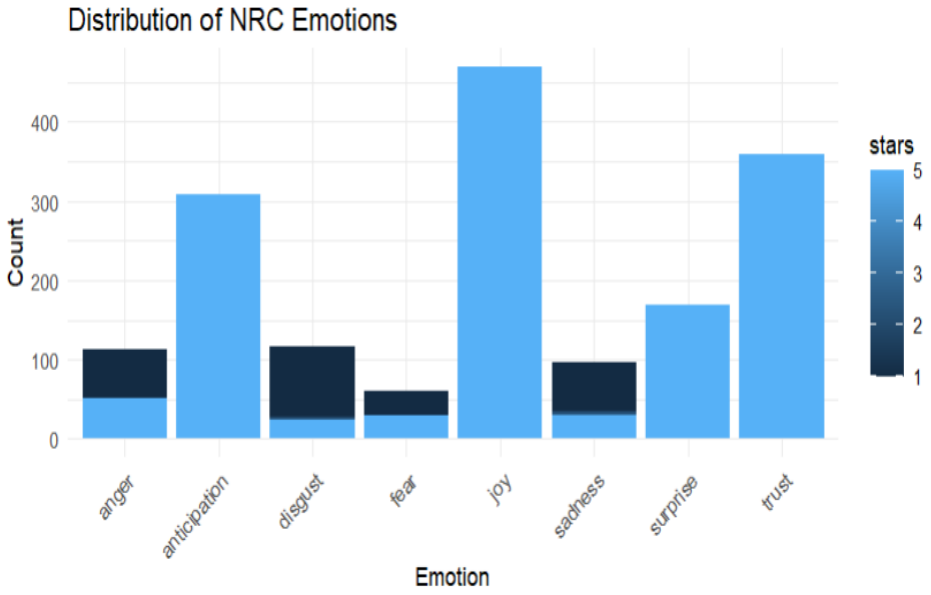
The above graph shows that

Neutral Sentiment Prevalence: The fifth histogram shows a significant peak at a score of 0, suggesting a high occurrence of neutral sentiment within the dataset.

Varied Sentiment Distribution: Histograms 1 and 2 exhibit flat distributions, indicating a more uniform spread of sentiment scores, while histograms 3 and 4 have slightly higher counts around the neutral score.

Overall, we can see that AFINN Sentiment Scores show positive sentiment for both Utilitarian and Hedonic products.

For utilitarian product (NRC)

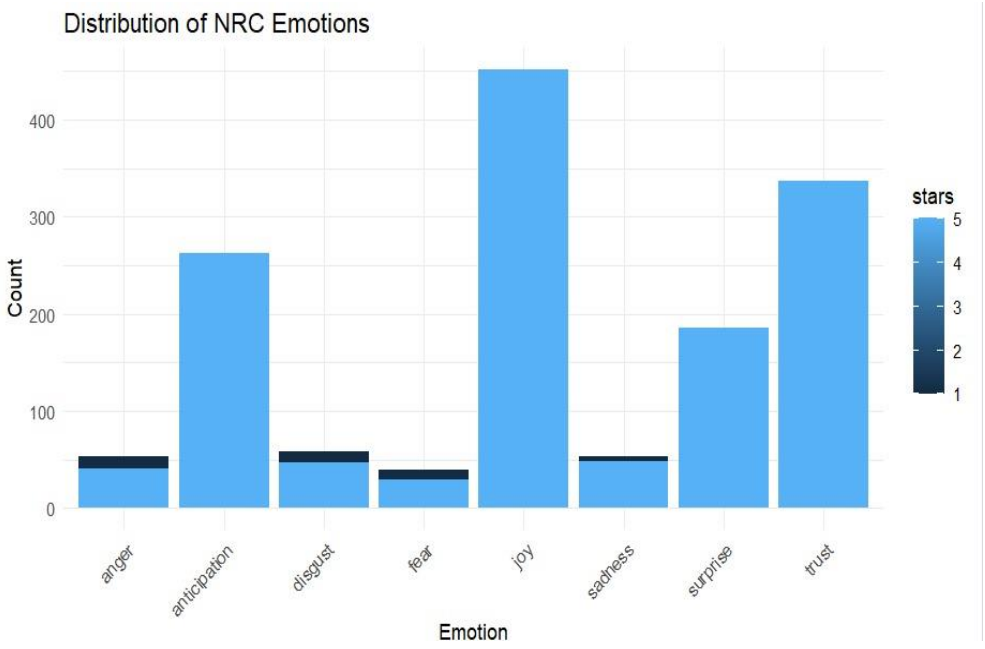


From the above graph we can see that

Dominant Emotions: The emotions joy and trust are the most frequently occurring in the dataset, with joy having the highest count of over 400 instances.

Lesser Emotions: Other emotions like anger, anticipation, disgust, fear, sadness, and surprise are represented significantly less.

For Hedonic Product (NRC)



From the above graph we can see that

Joy Dominates: The emotion of joy has the highest occurrence, with its count nearing 400.

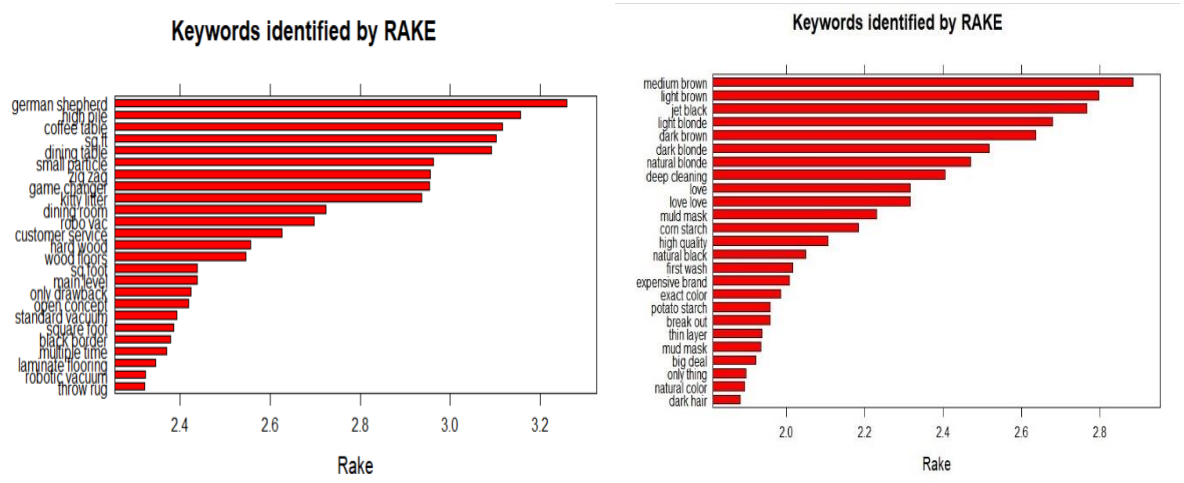
Trust and Anticipation: These emotions have moderate frequencies, with counts above 200 and approaching 300, respectively.

Lower Negative Emotions: Anger, disgust, fear, and sadness show significantly lower counts.

Surprise: This emotion has a moderate frequency, with its count above 100 but below 200.

Overall, we can see that there is positive sentiment for both utilitarian and hedonic product by analysing the AFINN Sentiment Scores and NRC distribution plot of emotions.

<u>Keywords extraction using RAKE:</u>



The graph on the left for utilitarian product includes a diverse range of keywords such as “german shepherd,” “coffee table,” and “wood vacuum,” suggesting the text analysed may have discussed topics from pet breeds to furniture and cleaning appliances.

The graph on the right for hedonic product shows that

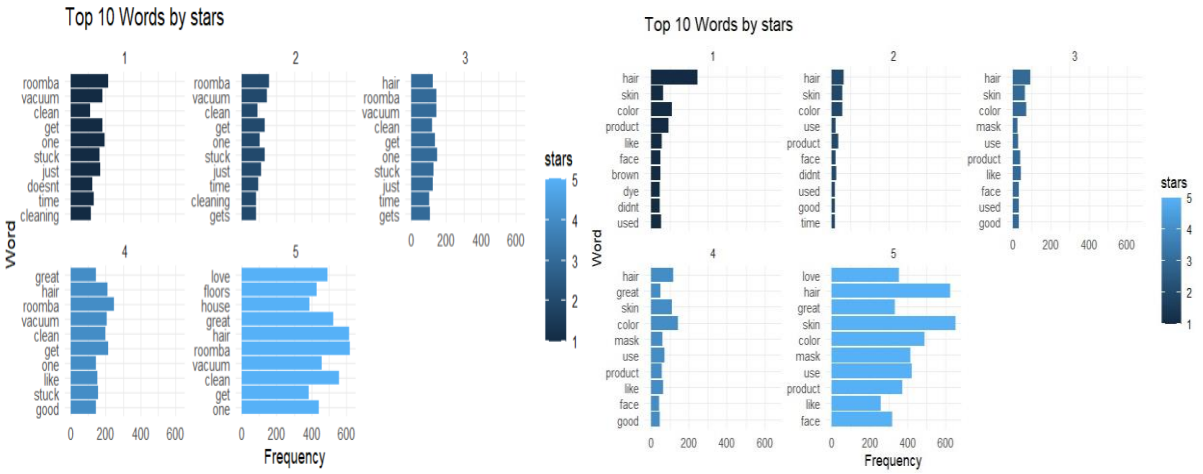
Popular Skin Colours: The keywords “medium brown,” “light brown,” “soft black,” “dark blonde,” and “natural blonde” suggest these are skin colours for the face mask.

Skin Care Trends: Terms like “deep cleaning,” “mud mask,” and “high quality” indicate a focus on intensive skin care treatments and premium products.

Consumer Preferences: The presence of keywords such as “love,” “colour stash,” and “natural colour” reflects a preference for natural looks and personalization on face.

These insights align with the current market trends that emphasize natural ingredients, sustainable practices, and personalized skin care solutions.

Term frequencies:



For utilitarian product

```
> print(head(term_freq))
# A tibble: 6 × 2
  word       n
  <chr>  <int>
1 roomba  1385
2 vacuum  1132
3 hair    1116
4 clean   1082
5 time     818
6 stuck    793
```

Top Concerns: The most frequently mentioned words in reviews for iRobot (Roomba) are "roomba," "vacuum," "hair," "clean," "time," and "stuck." This suggests that consumers are primarily concerned with aspects related to the cleaning performance and functionality of the vacuum robot.

Functional Features: Words like "clean" and "vacuum" indicate that consumers are likely discussing the effectiveness and efficiency of the cleaning process. They may be emphasizing how well the iRobot device removes dirt, pet hair, and other debris from floors.

Issues and Challenges: The presence of words like "stuck" suggests that some consumers may be experiencing issues or challenges with the device getting stuck in certain areas or encountering obstacles during operation. This could be a pain point that consumers expect the product to address or improve upon.

For Hedonic Product

```
> print(head(term_freq))
# A tibble: 6 × 2
  word       n
  <chr>  <int>
1 hair    1147
2 skin     952
3 color    873
```

```
4 product   604
5 mask      555
6 love       390
```

Skin and Appearance: The top terms for face masks include "hair," "skin," "colour," "product," "mask," and "love." This indicates that consumers are focused on aspects related to skincare and appearance enhancement.

Beauty Benefits: Words like "hair" and "skin" suggest that consumers are discussing the impact of the face mask on their skin's health and appearance. They may be emphasizing benefits such as hydration, cleansing, or improving skin texture.

Aesthetic Preferences: Terms like "colour" and "product" imply that consumers are interested in the visual and sensory aspects of the face mask, such as its colour, texture, scent, or packaging. These factors likely contribute to their overall enjoyment and satisfaction with the product.

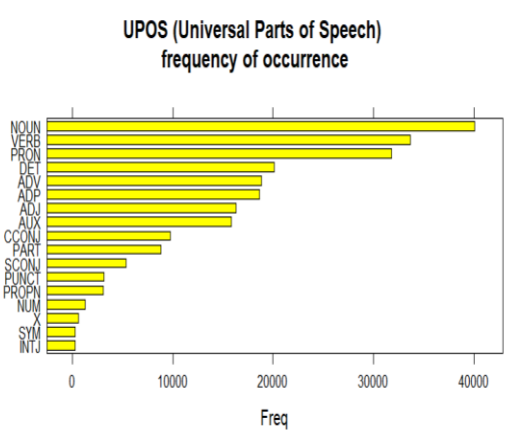
Positive Sentiment: The presence of the word "love" indicates that many consumers express strong positive sentiment towards the face mask. This suggests that the product is well-received and valued for its effectiveness or the experience it provides.

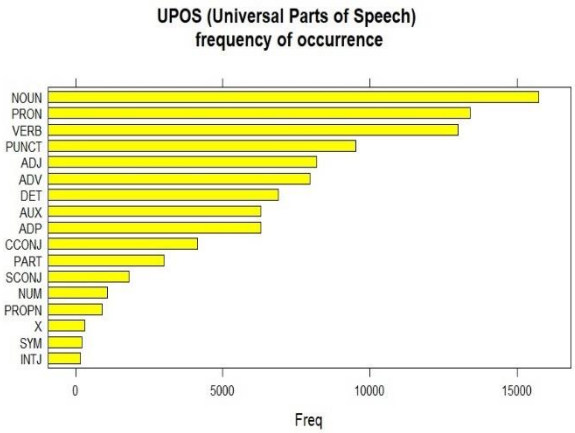
Marketing and Product Insights:

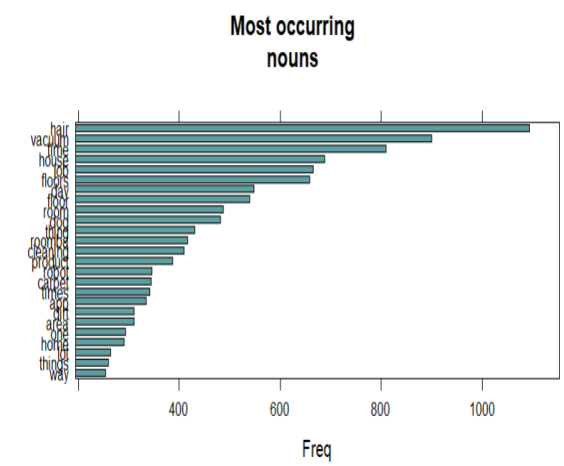
Utilitarian Product: Marketers and developers of iRobot can focus on highlighting the cleaning performance and efficiency of the device, addressing any common issues such as getting stuck, and emphasizing its practical benefits for consumers.

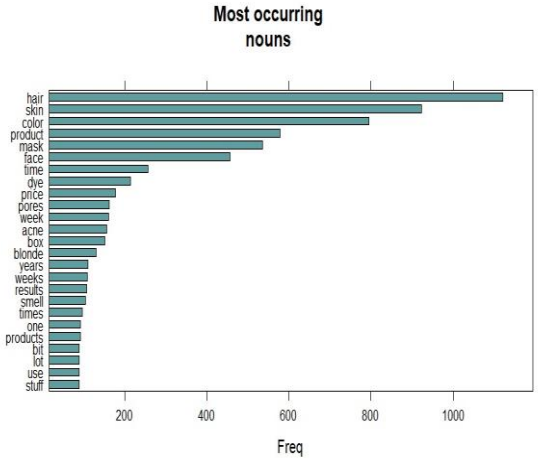
Hedonic Product: For face masks, marketing efforts can emphasize the skincare benefits, visual appeal, and sensory experience of using the product. Brands can leverage positive consumer sentiment and focus on creating an enjoyable and indulgent experience for users.

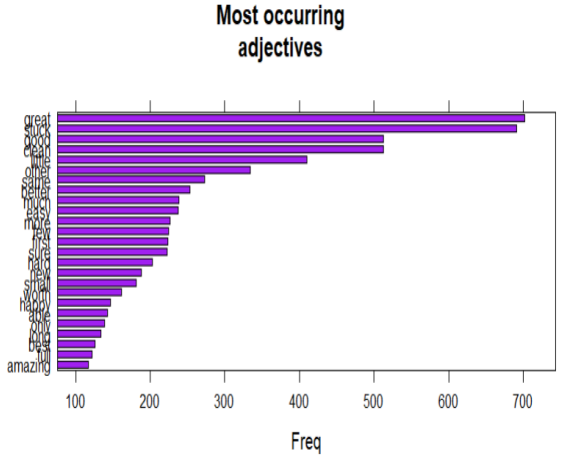
Parts of speech:

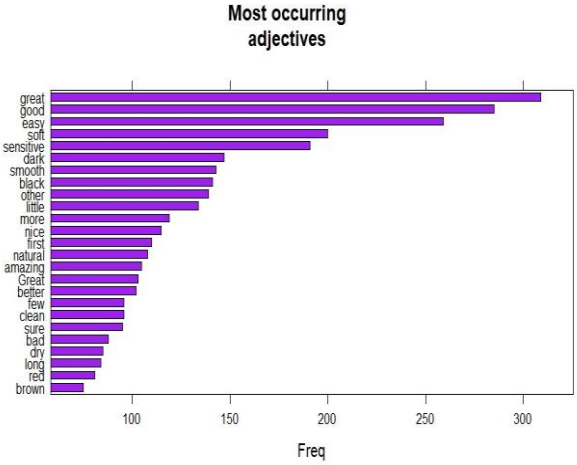


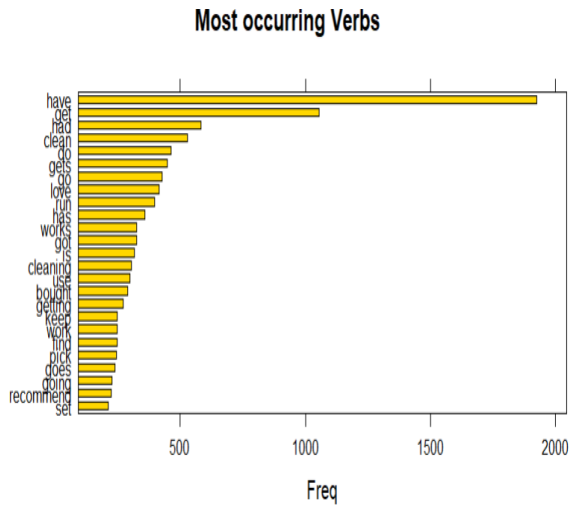


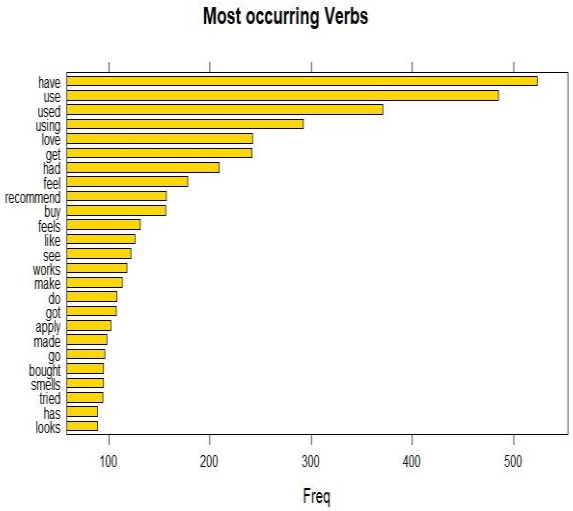












The graphs above in the left shows the parts of speech for utilitarian product. From the above graphs we can see that nouns are most frequently used for comments in utilitarian product. Also, most frequently used Noun is “hair,” most frequently used adjective is “great” and most frequently used Verb is “have.”

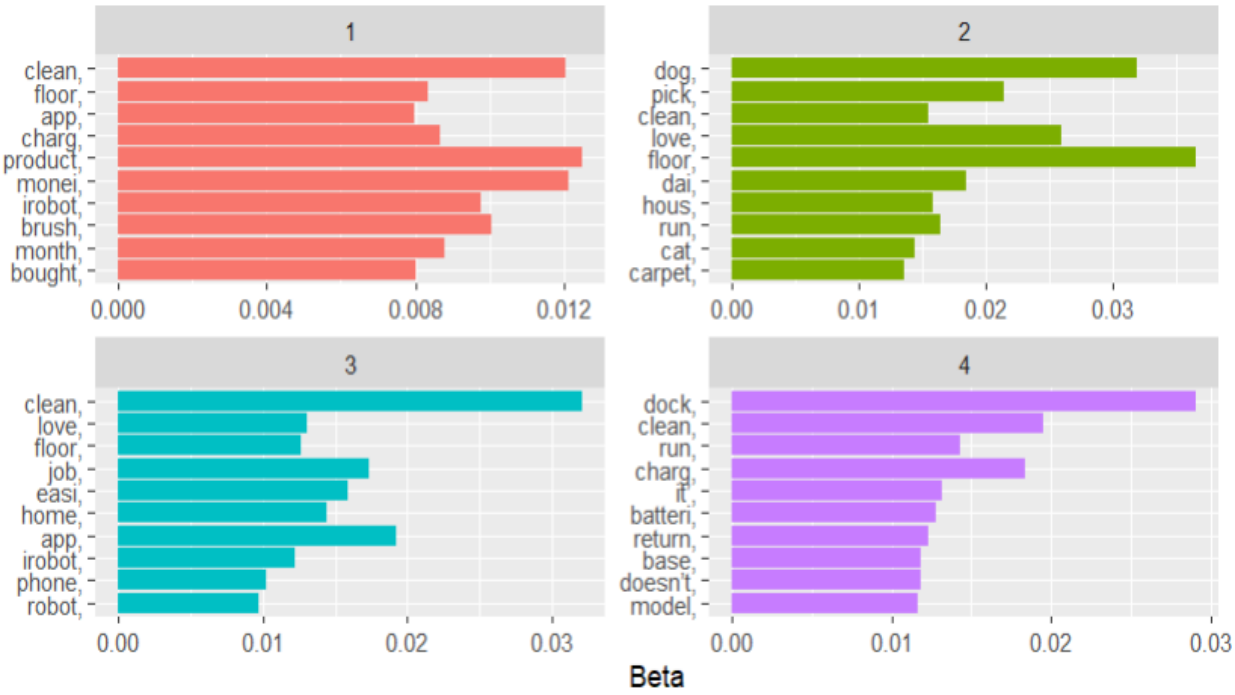
The graphs above in the right shows the parts of speech for hedonic product. From the above graphs we can see that nouns are most frequently used in fake news. Also, most frequently used

Noun is “hair,” most frequently used adjective is “great” and most frequently used Verb is “have.”

Overall, we can see that same kind of parts of speech are used in comments for both the products.

Topic modelling using LDA:

For utilitarian product



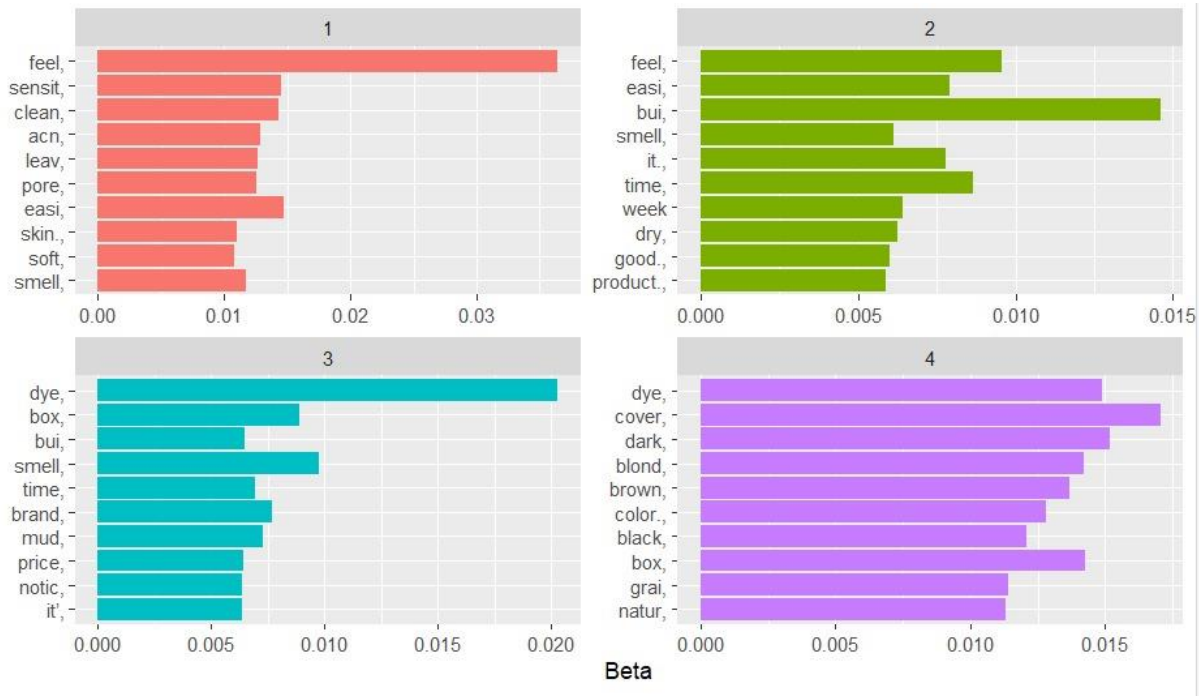
The above graph gives topic model using LDA for utilitarian product. We can see that each graph represents each different topic

Category Analysis: Each graph seems to represent a unique category, with words that are likely relevant to that specific context. For instance, words like “clean,” “floor,” and “app” suggest a category related to cleaning or maintenance.

Word Frequency: The bar lengths indicate the Beta values, which could be a measure of how frequently these words are mentioned or how significant they are within their respective categories.

Common Themes: Some words like “clean” appear in multiple categories, hinting at common themes or overlapping topics between different categories.

For Hedonic Product



The above graph gives topic model using LDA for hedonic product. We can see that each graph represents each different topic

Sensory Experience: Topic 1 (Red) highlights aspects related to the sensory experience of using the face mask, such as feelings and smells, which could be important for user satisfaction.

Ease of Use: Topic 2 (Green) suggests a focus on the face mask's ease of use and time efficiency, indicating that consumers value quick and simple solutions.

Product Details: Topic 3 (Blue) is associated with the product's packaging, price, and effects after use, which are crucial factors in the purchasing decision.

Variety and Aesthetics: Topic 4 (Purple) emphasizes the range of colours available and their compatibility with different skin tones, showing an emphasis on personalization and aesthetic appeal.

Star Rating Variances:

```
> print(paste("Variance of star ratings for nyb:", nyb_variance))
[1] "Variance of star ratings for nyb: 2.0100727863932"
> print(paste("Variance of star ratings for irobot:", irobot_variance))
[1] "Variance of star ratings for irobot: 2.23838601756141"
> # Compare variances
> if (nyb_variance > irobot_variance) {
+   print("Variance of star ratings is higher for hedonic product(nyb).")
+ } else if (nyb_variance < irobot_variance) {
+   print("Variance of star ratings is higher for utilitarian product(irob
ot).")
+ } else {
```

```
+   print("Variance of star ratings is equal for both product categories."
)
+ }
[1] "Variance of star ratings is higher for utilitarian product(irobot)."
```

Based on the above comparison of the star rating variances of the hedonic product "nyb" and the utilitarian product "irobot":

- The variance of star ratings for the "irobot" utilitarian product (2.238) is greater than that of the "nyb" hedonic product (2.010). This suggests that the utilitarian product has greater variability in star ratings than the hedonic product.
- The higher variance in star ratings for the utilitarian product "irobot" indicates that customers have more diverse perspectives and experiences with this product than the hedonic product "nyb".
- **Diverse Customer Opinions:** The higher variety in star ratings for the utilitarian product indicates that customers have different experiences and perspectives on the "irobot" product. This diversity could be attributed to product features, performance, and consumer expectations.
- **Consistency in Reviews:** The decreased variance in star ratings for the hedonic product "nyb" suggests a very regular trend in consumer feedback. This could imply that customers have more consistent experiences or views of the "nyb" product, which could be related to its intended use case or target population.
- **Marketing Considerations:** Marketers dealing with utilitarian items like "irobot" may need to pay more attention to consumer feedback and address any issues or concerns mentioned by a wide spectrum of customers. In contrast, marketers of hedonic items such as "nyb" may prioritise sustaining the positive sentiment and consistent customer experiences indicated in evaluations.

3) What are the key insights to know for marketers who deal with hedonic products and their reviews vs those dealing with utilitarian products?

<u>For Hedonic Product</u>

Emotion-driven Reviews: Hedonic products often evoke strong emotional responses in consumers, reflected in their reviews. Marketers should pay attention to the emotional language used by customers to understand their experiences and preferences.

Focus on Experience and Enjoyment: Consumers of hedonic products prioritize the experience and enjoyment derived from the product rather than just its practical utility. Marketers should emphasize these aspects in their messaging and branding to resonate with their target audience.

Brand Image and Lifestyle: Hedonic products are often associated with a lifestyle or aspirational image. Marketers should focus on building a brand identity that aligns with the lifestyle and values of their target consumers to foster brand loyalty and engagement.

Storytelling and Engagement: Leveraging storytelling techniques in marketing campaigns can create deeper connections with consumers of hedonic products by tapping into their emotions and aspirations.

For Utilitarian Product

Functionality and Practicality: Reviews of utilitarian products typically focus on functionality, reliability, and practicality rather than emotional experiences. Marketers should highlight these attributes in their messaging to appeal to consumers' rational decision-making process.

Problem-Solving Solutions: Consumers of utilitarian products seek solutions to specific problems or needs. Marketers should highlight how their product addresses these pain points and provides tangible benefits to users.

Value Proposition and ROI: Utilitarian products are evaluated based on their cost-effectiveness and return on investment. Marketers should emphasize the value proposition of their product, highlighting features that deliver tangible benefits and justify the price.

Rational Appeals and Information: Utilitarian product reviews tend to be more factual and information-driven. Marketers should provide clear and detailed information about product specifications, features, and performance to aid consumers in their decision-making process.

In both cases, positive sentiment in reviews can serve as valuable social proof and endorsement for the product, influencing other consumers' purchasing decisions. Marketers should actively monitor and leverage positive reviews to build brand credibility, enhance reputation, and drive sales. Additionally, addressing any negative feedback or concerns raised in reviews can help maintain customer satisfaction and trust in the long term.

R CODE

For Utilitarian Product

- Install and load required libraries

```
if (!require("tidyverse")) install.packages("tidyverse")

if (!require("data.table")) install.packages("data.table")

if (!require("tm")) install.packages("tm")

if (!require("textclean")) install.packages("textclean")

if (!require("tidytext")) install.packages("tidytext")

if (!require("wordcloud")) install.packages("wordcloud")

if (!require("SnowballC")) install.packages("SnowballC")

if (!requireNamespace("stringr", quietly = TRUE)) install.packages("stringr")

library(dplyr)

library(stringr)
```

```
library(textclean)
library(tidyverse)
library(data.table)
library(tm)
library(tidytext)
library(wordcloud)
library(SnowballC)
library(ggplot2)
```

- Load the dataset

```
irobot_data <- read.csv(file.choose(), stringsAsFactors = F)
analysis_data <- irobot_data %>% select(comments,stars)
```

- Clean and preprocess the text data

```
clean_text <- function(text) {
  text %>%
    tolower() %>%
    replace_non_ascii() %>%
    removeNumbers() %>%
    removePunctuation() %>%
    stripWhitespace()
}
analysis_data$comments<- sapply(analysis_data$comments, clean_text)
```

- Tokenizing the data and removing stopwords

```
analysis_data <- analysis_data %>%
  unnest_tokens(word,comments) %>%
  anti_join(get_stopwords(), by = "word")
```

- Calculate top 10 words by ratings

```
library(data.table)
# Convert analysis_data to a data.table
analysis_data <- as.data.table(analysis_data)
top_words <- analysis_data[, .(Count = .N), by = .(word, stars)]
```

```
top_words <- top_words[, head(.SD[order(-Count)], 10), by = stars]
```

- Plotting the top 10 words by ratings

```
ggplot(top_words, aes(x = reorder(word, Count), y = Count, fill = stars)) +
  geom_col() +
  coord_flip() +
  labs(x = "Word", y = "Frequency", title = "Top 10 Words by stars") +
  facet_wrap(~stars, scales = "free_y") +
  theme_minimal()
```

- Count word frequencies

```
word_frequencies <- analysis_data %>%
  count(word, sort = TRUE)
```

- Plotting top 20 words

```
ggplot(word_frequencies[1:20, ], aes(x = reorder(word, n), y = n)) +
  geom_col(fill = "dodgerblue") +
  coord_flip() +
  labs(x = "Word", y = "Frequency", title = "Top 20 Most Frequent Words")
```

- Check the frequency of each rating

```
stars_distribution <- irobot_data %>%
  count(stars, sort = TRUE)
```

- Plot ratings distribution

```
ggplot(stars_distribution, aes(x = stars, y = n, fill = stars)) +
  geom_bar(stat = "identity") +
  labs(title = "Distribution of Ratings", x = "Ratings", y = "Count")
```

- Word cloud for irobot product

```
irobot_words <- subset(analysis_data)
wordcloud(words = irobot_words$word, max.words = 100, random.order = FALSE, colors =
        brewer.pal(8, "Dark2"))
library(tidyr)
library(dplyr)
library(tidytext)
```

```
library(ggplot2)
```

- Tokenize the text and remove stopwords

```
tokens <- analysis_data %>%
  unnest_tokens(output = word, input = comments) %>%
  anti_join(get_stopwords())
```

- Create bigrams and trigrams

```
bigrams <- tokens %>%
  mutate(next_word = lead(word)) %>%
  filter(!is.na(next_word)) %>%
  unite(bigram, word, next_word, sep = " ")
trigrams <- tokens %>%
  mutate(next_word1 = lead(word),
         next_word2 = lead(word, 2)) %>%
  filter(!is.na(next_word1) & !is.na(next_word2)) %>%
  unite(trigram, word, next_word1, next_word2, sep = " ")
```

- Count the frequency of bigrams and trigrams by stars

```
bigram_frequency_by_stars <- bigrams %>%
  count(stars, bigram) %>%
  arrange(stars, desc(n))
trigram_frequency_by_stars <- trigrams %>%
  count(stars, trigram) %>%
  arrange(stars, desc(n))
```

- Plot the top 20 bigrams by ratings

```
ggplot(bigram_frequency_by_stars %>% group_by(stars) %>% top_n(20, n), aes(x = reorder(bigram, n), y = n, fill = stars)) +
  geom_col() +
  coord_flip() +
  facet_wrap(~stars, scales = "free_y") +
  labs(x = "Bigram", y = "Frequency", title = "Top 20 Bigrams by Ratings") +
  theme_minimal()
```

- Plot the top 10 trigrams by ratings

```
ggplot(trigram_frequency_by_stars %>% group_by(stars) %>% top_n(10, n), aes(x = reorder(trigram, n), y = n, fill = stars)) +
  geom_col() +
  coord_flip() +
  facet_wrap(~stars, scales = "free_y") +
  labs(x = "Trigram", y = "Frequency", title = "Top 10 Trigrams by Ratings") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 8))  # Adjust y-axis text size for better readability
```

Average length of the text:

- Load libraries

```
library(tidyverse)
library(tidytext)
```

- Read data

```
irobot_data <- read.csv(file.choose())
```

- Check column names in irobot_data

```
colnames(irobot_data)
```

- Check data structure of irobot_data

```
class(irobot_data)
```

- Check first few rows of irobot_data

```
head(irobot_data)
```

- Assuming 'comments' contains the text data

```
# Create tibble
irobot_tibble <- tibble(comments = irobot_data$comments)
```

- Tokenize the text

```
tokens <- irobot_tibble %>%
  mutate(document = row_number()) %>%
  unnest_tokens(word, comments)
```

- Count the number of words in each page

```
comment_lengths <- tokens %>%
  group_by(document) %>%
  summarise(num_words = n())
```

- Calculate the average length of the text

```
average_length <- mean(comment_lengths$num_words)
```

- Print the average length

```
print(paste("Average length of the text:", round(average_length, 2)))
```

Term frequencies:

- Load libraries

```
library(tidyverse)
library(tidytext)
```

- Read the dataset

```
irobot <- read.csv(file.choose())
irobot$comments <- str_replace_all(irobot$comments, "[^[:graph:]]", " ")
```

- Create a tibble

```
irobot_tibble <- tibble(title = irobot$comments)
```

- Tokenize the text

```
irobot_tokens <- irobot_tibble %>%
  mutate(document = row_number()) %>%
  unnest_tokens(word, title)
```

- Load stopwords list

```
data(stop_words)
```

- Remove stopwords

```
tokens_cleaned <- irobot_tokens %>%
  anti_join(stop_words, by = "word")
```

- Calculate term frequencies (after removing stopwords)

```
term_freq <- tokens_cleaned %>%
  count(word, sort = TRUE)
```

- Print the top terms by frequency

```
print(head(term_freq))
```

Parts of speech, Keywords extraction, Placement of words in general:

```
pacman::p_load(dplyr, ggplot2, stringr, udpipe, lattice)
irobot <- read.csv(file.choose(), stringsAsFactors = F)
```

```
udmodel_english <- udpipe_load_model(file = "english-ewt-ud-2.5-191206.udpipe")
s <- udpipe_annotate(udmodel_english,irobot$comments)
x <- data.frame(s)
stats <- txt_freq(x$upos)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = stats, col = "yellow",
        main = "UPOS (Universal Parts of Speech)\n frequency of occurrence",
        xlab = "Freq")
stats <- subset(x, upos %in% c("NOUN"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 25), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
stats <- subset(x, upos %in% c("ADJ"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 25), col = "purple", main = "Most occurring
adjectives", xlab = "Freq")
stats <- subset(x, upos %in% c("VERB"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 25), col = "gold", main = "Most occurring Verbs",
        xlab = "Freq")
stats <- keywords_rake(x = x, term = "lemma", group = "doc_id", relevant = x$upos %in%
                    c("NOUN", "ADJ"))
stats$key <- factor(stats$keyword, levels = rev(stats$keyword))
barchart(key ~ rake, data = head(subset(stats, freq > 3), 25), col = "red", main =
          "Keywords identified by RAKE", xlab = "Rake")
x$phrase_tag <- as_phrasemachine(x$upos, type = "upos")
stats <- keywords_phrases(x = x$phrase_tag, term = tolower(x$token), pattern = "(A|
```

```
N)*N(P+D*(A|N)*N)*", is_regex = TRUE, detailed = FALSE)
stats <- subset(stats, ngram > 1 & freq > 3)
stats$key <- factor(stats$keyword, levels = rev(stats$keyword))
barchart(key ~ freq, data = head(stats, 25), col = "magenta", main = "Keywords - simple
noun phrases", xlab = "Frequency")
stats <- keywords_collocation(x = x, term = "token", group = c("doc_id", "paragraph_id",
                                                               "sentence_id"), ngram_max = 4)
stats <- cooccurrence(x = subset(x, upos %in% c("NOUN", "ADJ")), term = "lemma", group
                      = c("doc_id", "paragraph_id", "sentence_id"))
stats <- cooccurrence(x = x$lemma, relevant = x$upos %in% c("NOUN", "ADJ"))
stats <- cooccurrence(x = x$lemma, relevant = x$upos %in% c("NOUN", "ADJ"), skipgram =
                        2)
head(stats)
pacman::p_load(igraph, ggraph)
wordnetwork <- head(stats, 25)
wordnetwork <- graph_from_data_frame(wordnetwork)
ggraph(wordnetwork, layout = "fr") + geom_edge_link(aes(width = cooc, edge_alpha =
                                                          cooc), edge_colour = "red") +
  geom_node_text(aes(label = name), col = "darkgreen", size = 4) +
  theme_graph(base_family = "Arial Narrow") +
  theme(legend.position = "none") +
  labs(title = "Co-occurrences within 3 words distance", subtitle = "Nouns & Adjectives")
```

Sentiment Analysis:

- Load required packages, install if not already installed

```
if (!require("tidyverse")) install.packages("tidyverse")
if (!require("data.table")) install.packages("data.table")
if (!require("tidytext")) install.packages("tidytext")
if (!require("syuzhet")) install.packages("syuzhet")
```

- Load necessary libraries

```
library(dplyr)
```

```
library(tidyr)
library(tidytext)
library(ggplot2)
```

- Confirm the AFINN lexicon structure

```
afinn <- get_sentiments("afinn")
print(head(afinn))
```

- Prepare data by adding an identifier and tokenizing

```
irobot$document_id <- seq_len(nrow(irobot))
```

- Tokenizing and joining with AFINN

```
processed_data <- irobot %>%
  unnest_tokens(word, comments) %>%
  inner_join(afinn, by = "word")
```

- Check what the processed data contains

```
print(head(processed_data))
```

- If 'processed_data' is empty, it means no words matched the AFINN words after tokenization

```
if (nrow(processed_data) == 0) {
  cat("No matches found between your data's words and AFINN lexicon after tokenization.\n")
} else {
```

- Perform sentiment scoring

```
  title_sentiment_afinn <- processed_data %>%
    group_by(document_id) %>%
    summarise(afinn_score = sum(value, na.rm = TRUE), .groups = 'drop')
```

- Merge the sentiment scores back with the main data

```
  irobot <- left_join(irobot, title_sentiment_afinn, by = "document_id")
```

- Visualize the results

```
  ggplot(irobot, aes(x = afinn_score, fill = stars)) +
    geom_histogram(bins = 50, alpha = 0.7) +
    facet_wrap(~stars) +
    labs(title = "Distribution of AFINN Sentiment Scores", x = "AFINN Sentiment Score", y = "Count")
```

```
}
```

- Install and load required packages

```
if (!require("tidyverse")) install.packages("tidyverse")
if (!require("tidytext")) install.packages("tidytext")
if (!require("ggplot2")) install.packages("ggplot2")
library(dplyr)
library(tidytext)
library(ggplot2)
irobot$document_id <- seq_len(nrow(irobot))
#Load the NRC emotion lexicon
nrc_emotions <- get_sentiments("nrc") %>%
  filter(sentiment %in% c("anger", "anticipation", "disgust", "fear", "joy",
                          "sadness", "surprise", "trust"))
```

- Tokenizing titles and scoring emotions

```
title_emotions <- irobot %>%
  unnest_tokens(word, title) %>%
  inner_join(nrc_emotions, by = "word") %>%
  count(document_id, stars, sentiment) %>%
  group_by(stars, sentiment) %>%
  summarise(emotion_count = sum(n), .groups = 'drop')
```

- Check results

```
print(head(title_emotions))
```

- Plotting emotions by ratings

```
ggplot(title_emotions, aes(x = sentiment, y = emotion_count, fill =stars)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.9)) +
  labs(title = "Distribution of NRC Emotions ",x = "Emotion", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

Topic modelling using LDA:

```
library(tidyverse)
```

```
library(tidytext)
library(topicmodels)
library(tm)
library(SnowballC)
library(stringr)
irobot <- read.csv(file.choose())
irobot$comments <- str_replace_all(irobot$comments,"[^[:graph:]]", " ")
top_terms_by_topic_LDA <- function(input_text, # should be a column from a data frame
                                   plot = T, # return a plot? TRUE by defult
                                   number_of_topics = 4) # number of topics (4 by default)
{
  # create a corpus (type of object expected by tm) and document term matrix
  Corpus <- Corpus(VectorSource(input_text)) # make a corpus object
  DTM <- DocumentTermMatrix(Corpus) # get the count of words/document
  # remove any empty rows in our document term matrix (if there are any
  # we'll get an error when we try to run our LDA)
  unique_indexes <- unique(DTM$i) # get the index of each unique value
  DTM <- DTM[unique_indexes,] # get a subset of only those indexes
  # preform LDA & get the words/topic in a tidy text format
  lda <- LDA(DTM, k = number_of_topics, control = list(seed = 1234))
  topics <- tidy(lda, matrix = "beta")
  # get the top ten terms for each topic,
  # yes I made up the word informativeness
  top_terms <- topics %>% # take the topics data frame and..
    group_by(topic) %>% # treat each topic as a different group
    top_n(10, beta) %>% # get the top 10 most informative words
    ungroup() %>% # ungroup
    arrange(topic, -beta) # arrange words in descending informativeness
  # if the user asks for a plot (TRUE by default)
  if(plot == T){
```

```
    # plot the top ten terms for each topic in order
    top_terms %>% # take the top terms
      mutate(term = reorder(term, beta)) %>% # sort terms by beta value
      ggplot(aes(term, beta, fill = factor(topic))) + # plot beta by theme
      geom_col(show.legend = FALSE) + # as a bar plot
      facet_wrap(~ topic, scales = "free") + # which each topic in a separate plot
      labs(x = NULL, y = "Beta") + # no x label, change y label
      coord_flip() # turn bars sideways
  }else{
    # if the user does not request a plot
    # return a list of sorted terms instead
    return(top_terms)
  }
}
top_terms_by_topic_LDA(irobot$comments, number_of_topics = 2)
Corpus <- Corpus(VectorSource(irobot$comments))
DTM <- DocumentTermMatrix(Corpus)
DTM_tidy <- tidy(DTM)
custom_stop_words <- tibble(word = c("roomba", "vacuum","hair", "clean", "time",
                                     "stuck"))
DTM_tidy_cleaned <- DTM_tidy %>% # take our tidy dtm and...
  anti_join(stop_words, by = c("term" = "word")) %>% # remove English stopwords and...
  anti_join(custom_stop_words, by = c("term" = "word"))
cleaned_documents <- DTM_tidy_cleaned %>%
  group_by(document) %>%
  mutate(terms = toString(rep(term, count))) %>%
  select(document, terms) %>%
  unique()
head(cleaned_documents)
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 2)
```

```
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 3)
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 4)
DTM_tidy_cleaned <- DTM_tidy_cleaned %>%
  mutate(stem = wordStem(term))
cleaned_documents <- DTM_tidy_cleaned %>%
  group_by(document) %>%
  mutate(terms = toString(rep(stem, count))) %>%
  select(document, terms) %>%
  unique()
top_terms_by_topic_LDA(cleaned_documents$terms, number_of_topics = 3)
top_terms_by_topic_LDA(cleaned_documents$terms, number_of_topics = 4)
```

For Hedonic Product

- Install and load required libraries

```
library(dplyr)
library(stringr)
library(textclean)
library(tidyverse)
library(data.table)
library(tm)
library(tidytext)
library(wordcloud)
library(SnowballC)
library(ggplot2)
```

- Read CSV file with UTF-8 encoding

```
nyb_data <- read.csv(file.choose(), stringsAsFactors = FALSE, fileEncoding = "UTF-8")
analysis_data <- nyb_data %>% select(comments,stars)
```

- Clean and preprocess the text data

```
clean_text <- function(text) {
  text %>%
    tolower() %>%
```

```
  replace_non_ascii() %>%
  removeNumbers() %>%
  removePunctuation() %>%
  stripWhitespace()
}
analysis_data$comments<- sapply(analysis_data$comments, clean_text)
```

- Tokenizing the data and removing stopwords

```
analysis_data <- analysis_data %>%
 unnest_tokens(word,comments) %>%
 anti_join(get_stopwords(), by = "word")
```

- Calculate top 10 words by ratings

```
library(data.table)
# Convert analysis_data to a data.table
analysis_data <- as.data.table(analysis_data)
top_words <- analysis_data[, .(Count = .N), by = .(word, stars)]
top_words <- top_words[, head(.SD[order(-Count)], 10), by = stars]
```

- Plotting the top 10 words by ratings

```
ggplot(top_words, aes(x = reorder(word, Count), y = Count, fill = stars)) +
 geom_col() +
 coord_flip() +
 labs(x = "Word", y = "Frequency", title = "Top 10 Words by stars") +
 facet_wrap(~stars, scales = "free_y") +
 theme_minimal()
```

- Count word frequencies

```
word_frequencies <- analysis_data %>%
 count(word, sort = TRUE)
```

- Plotting top 20 words

```
ggplot(word_frequencies[1:20, ], aes(x = reorder(word, n), y = n)) +
 geom_col(fill = "dodgerblue") +
 coord_flip() +
```

```
  labs(x = "Word", y = "Frequency", title = "Top 20 Most Frequent Words")
```

- Check the frequency of each rating

```
stars_distribution <- nyb_data %>%
  count(stars, sort = TRUE)
```

- Plot rating distribution

```
ggplot(stars_distribution, aes(x = stars, y = n, fill = stars)) +
  geom_bar(stat = "identity") +
  labs(title = "Distribution of Ratings", x = "Ratings", y = "Count")
```

- Word cloud for hedonic product

```
nyb_words <- subset(analysis_data)
wordcloud(words = nyb_words$word, max.words = 100, random.order = FALSE, colors = brewer.pal(8,"Dark2"))
library(tidyr)
library(dplyr)
library(tidytext)
library(ggplot2)
str(analysis_data)

# Assuming 'comments' contains the text data
```

- Tokenize the text and remove stopwords

```
tokens <- analysis_data %>%
  unnest_tokens(output = word, input = word) %>%
  anti_join(get_stopwords())
```

- Create bigrams and trigrams

```
bigrams <- tokens %>%
  mutate(next_word = lead(word)) %>%
  filter(!is.na(next_word)) %>%
  unite(bigram, word, next_word, sep = " ")
trigrams <- tokens %>%
  mutate(next_word1 = lead(word),
         next_word2 = lead(word, 2)) %>%
```

```
  filter(!is.na(next_word1) & !is.na(next_word2)) %>%
  unite(trigram, word, next_word1, next_word2, sep = " ")
```

- Count the frequency of bigrams and trigrams by ratings

```
bigram_frequency_by_stars <- bigrams %>%
  count(stars, bigram) %>%
  arrange(stars, desc(n))
trigram_frequency_by_stars <- trigrams %>%
  count(stars, trigram) %>%
  arrange(stars, desc(n))
```

- Plot the top 20 bigrams by ratings

```
ggplot(bigram_frequency_by_stars %>% group_by(stars) %>% top_n(10, n), aes(x = reorder(bigram, n), y = n, fill = stars)) +
  geom_col() +
  coord_flip () +
  facet_wrap(~stars, scales = "free_y") +
  labs(x = "Bigram", y = "Frequency", title = "Top 20 Bigrams by Ratings") +
  theme_minimal()
```

- Plot the top 20 trigrams by ratings

```
ggplot(trigram_frequency_by_stars %>% group_by(stars) %>% top_n(9, n), aes(x = reorder(trigram, n), y = n, fill = stars)) +
  geom_col() +
  coord_flip() +
  facet_wrap(~stars, scales = "free_y") +
  labs(x = "Trigram", y = "Frequency", title = "Top 20 Trigrams by Ratings") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 8))  # Adjust y-axis text size for better readability
```

Average length of the text:

- Load libraries

```
library(tidyverse)
library(tidytext)
```

- Read data

```
nyb_data <- read.csv(file.choose())
```

- Check column names in nyb_data

```
colnames(nyb_data)
```

- Check data structure of nyb_data

```
class(nyb_data)
```

- Check first few rows of nyb_data

```
head(nyb_data)
```

- Assuming 'comments' contains the text data

```
# Create tibble
nyb_tibble <- tibble(comments = nyb_data$comments)
```

- Tokenize the text

```
tokens <- nyb_tibble %>%
  mutate(document = row_number()) %>%
  unnest_tokens(word, comments)
```

- Count the number of words in each page

```
comment_lengths <- tokens %>%
  group_by(document) %>%
  summarise(num_words = n())
```

- Calculate the average length of the text

```
average_length <- mean(comment_lengths$num_words)
```

- Print the average length

```
print(paste("Average length of the text:", round(average_length,2)))
```

Term frequencies:

- Load libraries

```
library(tidyverse)
library(tidytext)
```

- Read the dataset

```
nyb <- read.csv(file.choose())
nyb$comments <- str_replace_all(nyb$comments, "[^[:graph:]]", " ")
```

- Create a tibble

```
nyb_tibble <- tibble(title = nyb$comments)
```

- Tokenize the text

```
nyb_tokens <- nyb_tibble %>%
  mutate(document = row_number()) %>%
  unnest_tokens(word, title)
```

- Load stopwords list

```
data(stop_words)
```

- Remove stopwords

```
tokens_cleaned <- nyb_tokens %>%
  anti_join(stop_words, by = "word")
```

- Calculate term frequencies (after removing stopwords)

```
term_freq <- tokens_cleaned %>%
  count(word, sort = TRUE)
```

- Print the top terms by frequency

```
print(head(term_freq))
```

Parts of speech, Keywords extraction, Placement of words in general:

```
pacman::p_load(dplyr, ggplot2, stringr, udpipe, lattice)
nyb <- read.csv(file.choose(), stringsAsFactors = F)
udmodel_english <- udpipe_load_model(file = "english-ewt-ud-2.5-191206.udpipe")
s <- udpipe_annotate(udmodel_english,nyb$comments)
x <- data.frame(s)
stats <- txt_freq(x$upos)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = stats, col = "yellow",
        main = "UPOS (Universal Parts of Speech)\n frequency of occurrence",
        xlab = "Freq")
stats <- subset(x, upos %in% c("NOUN"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
```

```
barchart(key ~ freq, data = head(stats, 25), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
stats <- subset(x, upos %in% c("ADJ"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 25), col = "purple", main = "Most occurring
adjectives", xlab = "Freq")
stats <- subset(x, upos %in% c("VERB"))
stats <- txt_freq(stats$token)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 25), col = "gold", main = "Most occurring Verbs",
        xlab = "Freq")
stats <- keywords_rake(x = x, term = "lemma", group = "doc_id", relevant = x$upos %in%
                    c("NOUN", "ADJ"))
stats$key <- factor(stats$keyword, levels = rev(stats$keyword))
barchart(key ~ rake, data = head(subset(stats, freq > 3), 25), col = "red", main =
          "Keywords identified by RAKE", xlab = "Rake")
x$phrase_tag <- as_phrasemachine(x$upos, type = "upos")
stats <- keywords_phrases(x = x$phrase_tag, term = tolower(x$token),
                     pattern = "(A|N)N(P+D(A|N)N)", is_regex = TRUE, detailed = FALSE)
stats <- subset(stats, ngram > 1 & freq > 3)
stats$key <- factor(stats$keyword, levels = rev(stats$keyword))
barchart(key ~ freq, data = head(stats, 25), col = "magenta",
        main = "Keywords - simple noun phrases", xlab = "Frequency")
stats <- keywords_collocation(x = x, term = "token", group = c("doc_id", "paragraph_id",
                                                "sentence_id"), ngram_max = 4)
stats <- cooccurrence(x = subset(x, upos %in% c("NOUN", "ADJ")), term = "lemma", group
                = c("doc_id", "paragraph_id", "sentence_id"))
stats <- cooccurrence(x = x$lemma, relevant = x$upos %in% c("NOUN", "ADJ"))
stats <- cooccurrence(x = x$lemma, relevant = x$upos %in% c("NOUN", "ADJ"), skipgram =
```

2)

```
head(stats)
pacman::p_load(igraph, ggraph)
wordnetwork <- head(stats, 25)
wordnetwork <- graph_from_data_frame(wordnetwork)
ggraph(wordnetwork, layout = "fr") + geom_edge_link(aes(width = cooc, edge_alpha =
                                                        cooc), edge_colour = "red") +
  geom_node_text(aes(label = name), col = "darkgreen", size = 4) +
  theme_graph(base_family = "Arial Narrow") +
  theme(legend.position = "none") +
  labs(title = "Co-occurrences within 3 words distance", subtitle = "Nouns&Adjectives")
```

Sentiment Analysis:

- Load required packages, install if not already installed

```
if (!require("tidyverse")) install.packages("tidyverse")
if (!require("data.table")) install.packages("data.table")
if (!require("tidytext")) install.packages("tidytext")
if (!require("syuzhet")) install.packages("syuzhet")
```

- Load necessary libraries

```
library(dplyr)
library(tidyr)
library(tidytext)
library(ggplot2)
```

- Confirm the AFINN lexicon structure

```
afinn <- get_sentiments("afinn")
print(head(afinn))
```

- Prepare data by adding an identifier and tokenizing

```
nyb$document_id <- seq_len(nrow(nyb))
```

- Tokenizing and joining with AFINN

```
processed_data <- nyb %>%
  unnest_tokens(word, comments) %>%
```

```
  inner_join(afinn, by = "word")
```

- Check what the processed data contains

```
print(head(processed_data))
```

- If 'processed_data' is empty, it means no words matched the AFINN words after tokenization

```
if (nrow(processed_data) == 0) {
  cat("No matches found between your data's words and AFINN lexicon after tokenization.\n")
} else {
```

- Perform sentiment scoring

```
  title_sentiment_afinn <- processed_data %>%
    group_by(document_id) %>%
    summarise(afinn_score = sum(value, na.rm = TRUE), .groups = 'drop')
```

- Merge the sentiment scores back with the main data

```
  nyb <- left_join(nyb, title_sentiment_afinn, by = "document_id")
```

- Visualize the results

```
  ggplot(nyb, aes(x = afinn_score, fill = stars)) +
    geom_histogram(bins = 50, alpha = 0.7) +
    facet_wrap(~stars) +
    labs(title = "Distribution of AFINN Sentiment Scores", x = "AFINN Sentiment Score", y = "Count")
}
```

- Install and load required packages

```
if (!require("tidyverse")) install.packages("tidyverse")
if (!require("tidytext")) install.packages("tidytext")
if (!require("ggplot2")) install.packages("ggplot2")
library(dplyr)
library(tidytext)
library(ggplot2)
nyb$document_id <- seq_len(nrow(nyb))
```

- Load the NR C emotion lexicon

```
nrc_emotions <- get_sentiments("nrc") %>%
```

```
  filter(sentiment %in% c("anger", "anticipation", "disgust", "fear", "joy",
                          "sadness", "surprise", "trust"))
```

- Tokenizing titles and scoring emotions

```
title_emotions <- nyb %>%
  unnest_tokens(word, title) %>%
  inner_join(nrc_emotions, by = "word") %>%
  count(document_id, stars, sentiment) %>%
  group_by(stars, sentiment) %>%
  summarise(emotion_count = sum(n), .groups = 'drop')
```

- Check results

```
print(head(title_emotions))
```

- Plotting emotions by ratings

```
ggplot(title_emotions, aes(x = sentiment, y = emotion_count, fill =stars)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.9)) +
  labs(title = "Distribution of NRC Emotions ",x = "Emotion", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

Topic modelling using LDA:

```
library(tidyverse)
library(tidytext)
library(topicmodels)
library(tm)
library(SnowballC)
library(stringr)
nyb <- read.csv(file.choose())
nyb$comments <- str_replace_all(nyb$comments,"[^[:graph:]]", " ")
top_terms_by_topic_LDA <- function(input_text, # should be a column from a data frame
                                   plot = T, # return a plot? TRUE by defult
                                   number_of_topics = 4) # number of topics (4 by default)
{
```

```
# create a corpus (type of object expected by tm) and document term matrix
Corpus <- Corpus(VectorSource(input_text)) # make a corpus object
DTM <- DocumentTermMatrix(Corpus) # get the count of words/document
# remove any empty rows in our document term matrix (if there are any
# we'll get an error when we try to run our LDA)
unique_indexes <- unique(DTM$i) # get the index of each unique value
DTM <- DTM[unique_indexes,] # get a subset of only those indexes
# preform LDA & get the words/topic in a tidy text format
lda <- LDA(DTM, k = number_of_topics, control = list(seed = 1234))
topics <- tidy(lda, matrix = "beta")
# get the top ten terms for each topic,
# yes I made up the word informativeness
top_terms <- topics %>% # take the topics data frame and..
  group_by(topic) %>% # treat each topic as a different group
  top_n(10, beta) %>% # get the top 10 most informative words
  ungroup() %>% # ungroup
  arrange(topic, -beta) # arrange words in descending informativeness
# if the user asks for a plot (TRUE by default)
if(plot == T){
  # plot the top ten terms for each topic in order
  top_terms %>% # take the top terms
    mutate(term = reorder(term, beta)) %>% # sort terms by beta value
    ggplot(aes(term, beta, fill = factor(topic))) + # plot beta by theme
    geom_col(show.legend = FALSE) + # as a bar plot
    facet_wrap(~ topic, scales = "free") + # which each topic in a separate plot
    labs(x = NULL, y = "Beta") + # no x label, change y label
    coord_flip() # turn bars sideways
}else{
  # if the user does not request a plot
  # return a list of sorted terms instead
```

```
    return(top_terms)
  }
}
top_terms_by_topic_LDA(nyb$comments, number_of_topics = 2)
Corpus <- Corpus(VectorSource(nyb$comments))
DTM <- DocumentTermMatrix(Corpus)
DTM_tidy <- tidy(DTM)
custom_stop_words <- tibble(word = c("hair", "skin","hair", "color", "product", "mask", "love"))
DTM_tidy_cleaned <- DTM_tidy %>% # take our tidy dtm and...
  anti_join(stop_words, by = c("term" = "word")) %>% # remove English stopwords and...
  anti_join(custom_stop_words, by = c("term" = "word"))
cleaned_documents <- DTM_tidy_cleaned %>%
  group_by(document) %>%
  mutate(terms = toString(rep(term, count))) %>%
  select(document, terms) %>%
  unique()
head(cleaned_documents)
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 2)
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 3)
top_terms_by_topic_LDA(cleaned_documents, number_of_topics = 4)
DTM_tidy_cleaned <- DTM_tidy_cleaned %>%
  mutate(stem = wordStem(term))
cleaned_documents <- DTM_tidy_cleaned %>%
  group_by(document) %>%
  mutate(terms = toString(rep(stem, count))) %>%
  select(document, terms) %>%
  unique()
top_terms_by_topic_LDA(cleaned_documents$terms, number_of_topics = 3)
top_terms_by_topic_LDA(cleaned_documents$terms, number_of_topics = 4)
```

Star Rating Variances:

- Load the datasets

```
nyb_data <- read.csv(file.choose(), stringsAsFactors = FALSE, fileEncoding = "UTF-8")
```

- Calculate variance of star ratings for hedonic products

```
nyb_variance <- var(nyb_data$stars)
```

- Calculate variance of star ratings for utilitarian products

```
irobot_variance <- var(irobot_data$stars)
```

- Print the variances

```
print(paste("Variance of star ratings for nyb:", nyb_variance))
print(paste("Variance of star ratings for irobot:", irobot_variance))
```

- Compare variances

```
if (nyb_variance > irobot_variance) {
  print("Variance of star ratings is higher for hedonic product(nyb).")
} else if (nyb_variance < irobot_variance) {
  print("Variance of star ratings is higher for utilitarian product(irobot).")
} else {
  print("Variance of star ratings is equal for both product categories.")
}
```