R Notebook

BIS 581 - Product Sentiment Analysis

# load libraries that you feel you need and explain why

# Load the data from a file

# Assess the data

# Viewing the dataset  
View(posts)

# Viewing the structure of dataset  
str(posts)

## 'data.frame': 27669 obs. of 18 variables:  
## $ Transaction\_ID : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ TimeKey : chr "2018-1" "2018-1" "2018-1" "2018-1" ...  
## $ Year : int 2018 2018 2018 2018 2018 2018 2018 2018 2018 2018 ...  
## $ Week : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Number\_of\_Items : int 4 4 4 3 1 5 3 1 5 2 ...  
## $ T\_Ounces : int 24 36 36 24 4 24 16 8 32 8 ...  
## $ T\_Price : chr "$21.00 " "$23.50 " "$22.50 " "$17.00 " ...  
## $ T\_Cost : chr "$8.02 " "$11.71 " "$11.65 " "$7.86 " ...  
## $ CustomerID : int NA NA NA NA NA NA NA NA NA NA ...  
## $ Star\_Rating : int NA NA NA NA NA NA NA NA NA NA ...  
## $ Comment : chr "" "" "" "" ...  
## $ Chocolate : int 1 1 0 0 1 1 1 0 1 1 ...  
## $ Feature : int 2 2 3 0 0 1 1 1 3 1 ...  
## $ Superman : int 0 0 1 0 0 1 1 0 0 0 ...  
## $ Vanilla : int 1 1 0 3 0 2 0 0 1 0 ...  
## $ FeatureFlavor : chr "Superman" "Superman" "Superman" "Superman" ...  
## $ ParlorLocation : chr "Alpena" "Alpena" "Alpena" "Alpena" ...  
## $ ProductionLocation: chr "Mount Pleasant" "Mount Pleasant" "Mount Pleasant" "Mount Pleasant" ...

# Display the first few rows  
head(posts)

## Transaction\_ID TimeKey Year Week Number\_of\_Items T\_Ounces T\_Price T\_Cost  
## 1 1 2018-1 2018 1 4 24 $21.00 $8.02   
## 2 2 2018-1 2018 1 4 36 $23.50 $11.71   
## 3 3 2018-1 2018 1 4 36 $22.50 $11.65   
## 4 4 2018-1 2018 1 3 24 $17.00 $7.86   
## 5 5 2018-1 2018 1 1 4 $3.75 $1.35   
## 6 6 2018-1 2018 1 5 24 $22.75 $8.12   
## CustomerID Star\_Rating Comment Chocolate Feature Superman Vanilla  
## 1 NA NA 1 2 0 1  
## 2 NA NA 1 2 0 1  
## 3 NA NA 0 3 1 0  
## 4 NA NA 0 0 0 3  
## 5 NA NA 1 0 0 0  
## 6 NA NA 1 1 1 2  
## FeatureFlavor ParlorLocation ProductionLocation  
## 1 Superman Alpena Mount Pleasant  
## 2 Superman Alpena Mount Pleasant  
## 3 Superman Alpena Mount Pleasant  
## 4 Superman Alpena Mount Pleasant  
## 5 Superman Alpena Mount Pleasant  
## 6 Superman Alpena Mount Pleasant

# Is any data prep needed? If so, adjust data here and comment on why in your submission

# Cleaning T\_Price and T\_Cost columns by removing $ symbols and converting to numeric  
# To enable numerical analysis, as price and cost are currently stored as characters.  
posts\_clean <- posts %>%  
 mutate(  
 # Removing non-numeric characters for calculations.  
 T\_Price\_numeric = as.numeric(gsub("[^0-9.]", "", T\_Price)),   
 T\_Cost\_numeric = as.numeric(gsub("[^0-9.]", "", T\_Cost)),  
   
 # Converting back to character format with '$'.  
 T\_Price = paste0("$", format(T\_Price\_numeric, nsmall = 2)),   
 T\_Cost = paste0("$", format(T\_Cost\_numeric, nsmall = 2))   
 )  
  
  
# Filling missing values in numeric columns with 0. To avoid issues with calculations due to missing values and maintain data consistency.  
numeric\_columns <- c("Star\_Rating", "T\_Price\_numeric", "T\_Cost\_numeric")  
posts\_clean <- posts\_clean %>%  
 mutate(across(all\_of(numeric\_columns), ~ ifelse(is.na(.), 0, .)))  
  
# Filling missing values in the Comment column with 'Unknown' and convert text to lowercase  
posts\_clean <- posts\_clean %>%  
 mutate(  
 # Replacing missing and blank comments  
 Comment = ifelse(is.na(Comment) | Comment == "", "unknown", Comment),  
   
 # Replacing missing CustomerID with 0  
 CustomerID = ifelse(is.na(CustomerID), 0, CustomerID),  
   
 # Converting all character fields to lowercase  
 across(where(is.character), tolower)   
 )  
  
# Removing duplicate rows if any exist. To ensure each transaction is only counted once for accurate analysis.  
posts\_clean <- posts\_clean %>% distinct()  
  
str(posts\_clean)

## 'data.frame': 27669 obs. of 20 variables:  
## $ Transaction\_ID : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ TimeKey : chr "2018-1" "2018-1" "2018-1" "2018-1" ...  
## $ Year : int 2018 2018 2018 2018 2018 2018 2018 2018 2018 2018 ...  
## $ Week : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Number\_of\_Items : int 4 4 4 3 1 5 3 1 5 2 ...  
## $ T\_Ounces : int 24 36 36 24 4 24 16 8 32 8 ...  
## $ T\_Price : chr "$21.00" "$23.50" "$22.50" "$17.00" ...  
## $ T\_Cost : chr "$ 8.02" "$11.71" "$11.65" "$ 7.86" ...  
## $ CustomerID : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Star\_Rating : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Comment : chr "unknown" "unknown" "unknown" "unknown" ...  
## $ Chocolate : int 1 1 0 0 1 1 1 0 1 1 ...  
## $ Feature : int 2 2 3 0 0 1 1 1 3 1 ...  
## $ Superman : int 0 0 1 0 0 1 1 0 0 0 ...  
## $ Vanilla : int 1 1 0 3 0 2 0 0 1 0 ...  
## $ FeatureFlavor : chr "superman" "superman" "superman" "superman" ...  
## $ ParlorLocation : chr "alpena" "alpena" "alpena" "alpena" ...  
## $ ProductionLocation: chr "mount pleasant" "mount pleasant" "mount pleasant" "mount pleasant" ...  
## $ T\_Price\_numeric : num 21 23.5 22.5 17 3.75 ...  
## $ T\_Cost\_numeric : num 8.02 11.71 11.65 7.86 1.35 ...

head(posts\_clean)

## Transaction\_ID TimeKey Year Week Number\_of\_Items T\_Ounces T\_Price T\_Cost  
## 1 1 2018-1 2018 1 4 24 $21.00 $ 8.02  
## 2 2 2018-1 2018 1 4 36 $23.50 $11.71  
## 3 3 2018-1 2018 1 4 36 $22.50 $11.65  
## 4 4 2018-1 2018 1 3 24 $17.00 $ 7.86  
## 5 5 2018-1 2018 1 1 4 $ 3.75 $ 1.35  
## 6 6 2018-1 2018 1 5 24 $22.75 $ 8.12  
## CustomerID Star\_Rating Comment Chocolate Feature Superman Vanilla  
## 1 0 0 unknown 1 2 0 1  
## 2 0 0 unknown 1 2 0 1  
## 3 0 0 unknown 0 3 1 0  
## 4 0 0 unknown 0 0 0 3  
## 5 0 0 unknown 1 0 0 0  
## 6 0 0 unknown 1 1 1 2  
## FeatureFlavor ParlorLocation ProductionLocation T\_Price\_numeric  
## 1 superman alpena mount pleasant 21.00  
## 2 superman alpena mount pleasant 23.50  
## 3 superman alpena mount pleasant 22.50  
## 4 superman alpena mount pleasant 17.00  
## 5 superman alpena mount pleasant 3.75  
## 6 superman alpena mount pleasant 22.75  
## T\_Cost\_numeric  
## 1 8.02  
## 2 11.71  
## 3 11.65  
## 4 7.86  
## 5 1.35  
## 6 8.12

View(posts\_clean)

# Decide if you want to filter

# Filter out rows without meaningful comments or customer information  
# Filtering out rows where Comment is 'unknown' to focus on meaningful feedback.  
# Filtering out rows where CustomerID is '0' to ensure we can analyze customer behavior.  
posts\_filtered <- posts\_clean %>%  
 filter(Comment != "unknown", CustomerID != 0)  
str(posts\_filtered)

## 'data.frame': 13569 obs. of 20 variables:  
## $ Transaction\_ID : int 5057 5058 5059 5061 5066 5069 5075 5076 5077 5078 ...  
## $ TimeKey : chr "2019-1" "2019-1" "2019-1" "2019-1" ...  
## $ Year : int 2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...  
## $ Week : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Number\_of\_Items : int 2 2 3 3 3 6 4 3 3 5 ...  
## $ T\_Ounces : int 8 16 16 16 20 32 36 12 16 32 ...  
## $ T\_Price : chr "$ 8.50" "$10.25" "$15.25" "$15.25" ...  
## $ T\_Cost : chr "$ 2.69" "$ 5.20" "$ 5.42" "$ 5.42" ...  
## $ CustomerID : num 903 959 5 1123 1375 ...  
## $ Star\_Rating : num 4 5 5 4 4 5 3 5 5 5 ...  
## $ Comment : chr "if you like cherries this one's for you" "ice cream is my love language" "when life gives you lemons, make a lemon sorbet to go with your ice cream" "superman you are flying high" ...  
## $ Chocolate : int 1 0 1 0 0 1 0 0 0 1 ...  
## $ Feature : int 0 2 2 2 3 4 2 1 2 1 ...  
## $ Superman : int 0 0 0 1 0 1 1 0 0 0 ...  
## $ Vanilla : int 1 0 0 0 0 0 1 2 1 3 ...  
## $ FeatureFlavor : chr "superman" "superman" "superman" "superman" ...  
## $ ParlorLocation : chr "alpena" "alpena" "alpena" "alpena" ...  
## $ ProductionLocation: chr "mount pleasant" "mount pleasant" "mount pleasant" "mount pleasant" ...  
## $ T\_Price\_numeric : num 8.5 10.2 15.2 15.2 15 ...  
## $ T\_Cost\_numeric : num 2.69 5.2 5.42 5.42 6.61 ...

head(posts\_filtered)

## Transaction\_ID TimeKey Year Week Number\_of\_Items T\_Ounces T\_Price T\_Cost  
## 1 5057 2019-1 2019 1 2 8 $ 8.50 $ 2.69  
## 2 5058 2019-1 2019 1 2 16 $10.25 $ 5.20  
## 3 5059 2019-1 2019 1 3 16 $15.25 $ 5.42  
## 4 5061 2019-1 2019 1 3 16 $15.25 $ 5.42  
## 5 5066 2019-1 2019 1 3 20 $15.00 $ 6.61  
## 6 5069 2019-1 2019 1 6 32 $27.25 $10.65  
## CustomerID Star\_Rating  
## 1 903 4  
## 2 959 5  
## 3 5 5  
## 4 1123 4  
## 5 1375 4  
## 6 642 5  
## Comment  
## 1 if you like cherries this one's for you  
## 2 ice cream is my love language  
## 3 when life gives you lemons, make a lemon sorbet to go with your ice cream  
## 4 superman you are flying high  
## 5 so good it's bad. what an ice cream shop  
## 6 capturing the essence of indulgence in every scoop  
## Chocolate Feature Superman Vanilla FeatureFlavor ParlorLocation  
## 1 1 0 0 1 superman alpena  
## 2 0 2 0 0 superman alpena  
## 3 1 2 0 0 superman alpena  
## 4 0 2 1 0 superman alpena  
## 5 0 3 0 0 superman alpena  
## 6 1 4 1 0 superman alpena  
## ProductionLocation T\_Price\_numeric T\_Cost\_numeric  
## 1 mount pleasant 8.50 2.69  
## 2 mount pleasant 10.25 5.20  
## 3 mount pleasant 15.25 5.42  
## 4 mount pleasant 15.25 5.42  
## 5 mount pleasant 15.00 6.61  
## 6 mount pleasant 27.25 10.65

# Define a list of common ice cream-related words and phrases to remove  
remove\_words <- c("ice", "cream", "vanilla", "chocolate", "strawberry",   
 "flavor", "cone", "scoop", "dessert", "topping",   
 "milkshake", "gelato", "sundae", "frozen", "custard",   
 "i'm", "too") # Added "i'm" and "too"  
  
# 🔥 Step 1: Tokenize Comments and Remove Unwanted Words  
cleaned\_comments <- posts\_filtered %>%  
 unnest\_tokens(word, Comment) %>%  
 filter(!word %in% remove\_words) %>%  
 group\_by(CustomerID) %>%  
 summarize(Clean\_Comment = paste(word, collapse = " "), .groups = "drop")  
  
# 🔥 Step 2: Merge Cleaned Comments Back Into Original Dataset  
posts\_filtered\_cleaned <- posts\_filtered %>%  
 select(-Comment) %>% # Remove original Comment column  
 left\_join(cleaned\_comments, by = "CustomerID") %>% # Merge cleaned comments  
 rename(Comment = Clean\_Comment) # Rename back to Comment  
  
# 🔥 Step 3: Ensure All Columns Are Present  
head(posts\_filtered\_cleaned)

## Transaction\_ID TimeKey Year Week Number\_of\_Items T\_Ounces T\_Price T\_Cost  
## 1 5057 2019-1 2019 1 2 8 $ 8.50 $ 2.69  
## 2 5058 2019-1 2019 1 2 16 $10.25 $ 5.20  
## 3 5059 2019-1 2019 1 3 16 $15.25 $ 5.42  
## 4 5061 2019-1 2019 1 3 16 $15.25 $ 5.42  
## 5 5066 2019-1 2019 1 3 20 $15.00 $ 6.61  
## 6 5069 2019-1 2019 1 6 32 $27.25 $10.65  
## CustomerID Star\_Rating Chocolate Feature Superman Vanilla FeatureFlavor  
## 1 903 4 1 0 0 1 superman  
## 2 959 5 0 2 0 0 superman  
## 3 5 5 1 2 0 0 superman  
## 4 1123 4 0 2 1 0 superman  
## 5 1375 4 0 3 0 0 superman  
## 6 642 5 1 4 1 0 superman  
## ParlorLocation ProductionLocation T\_Price\_numeric T\_Cost\_numeric  
## 1 alpena mount pleasant 8.50 2.69  
## 2 alpena mount pleasant 10.25 5.20  
## 3 alpena mount pleasant 15.25 5.42  
## 4 alpena mount pleasant 15.25 5.42  
## 5 alpena mount pleasant 15.00 6.61  
## 6 alpena mount pleasant 27.25 10.65  
## Comment  
## 1 if you like cherries this one's for you yes the candyland can the of champions sorry rocky road i found a new favorite better than most i’m here for a good time and by good time i mean time a masterpiece in every serving a delight in every bite i like my in a waffle and my days in sprinkles this is crazy whoever thought of this this is crazy whoever thought of this waited for ever to get served but it was worth it willing to scale a tall mountain for this cookie can i buy a truck load my life may not be perfect but my sure is mellowing out with soothing saskatoon a that transcends generations and brings smiles there’s nothing a of can’t fix why have abs when you can have make sundaes not war in the world of there are no calories there’s no such thing as much willing to walk the mighty mac for mighty mac fudge because every day is a special occasion eating is like a hug from the inside cherishing the moments made sweeter with a that transcends generations and brings smiles  
## 2 is my love language a that never goes out of style a that transcends time and place hello heaven where have you been all my life i like my in a waffle and my days in sprinkles wow you can’t buy love but you can buy and that’s basically the same thing makes everything better trust me i’m an expert this is crazy whoever thought of this makes everything better trust me i’m an expert cherry pie was ok is the cherry on top of life is my therapy better than most sweet dreams are made of this life’s short not to indulge in elegantly plated perfectly enjoyed really good to eat or not to eat is that the question i can't get enough i prefer the cherries i can't get enough my head says gym but my heart says  
## 3 when life gives you lemons make a lemon sorbet to go with your elevating the ordinary to the extraordinary with cherishing the moments made sweeter with  
## 4 superman you are flying high  
## 5 so good it's bad what an shop  
## 6 capturing the essence of indulgence in every good in the world of desserts is the crown jewel i’m not emotional i just need to survive thimbleberry thunder will rock your world if you like cherries this one's for you i believe in love at first bite of an ode to the timeless joy of you’re the cherry on top of my i like my in a waffle and my days in sprinkles life is better with a cherry on top elevating the ordinary to the extraordinary with simply cherry delighting my way through this there is no mint mystery about it if you like mint you need to get this feature sweet dreams are made of this i didn't know you could make this good there’s no we in it’s all me if you can’t handle me at my creamiest you don’t deserve me at my salad the only thing that solves problems without any side effects sure tempt me with that crazy glad it only comes once per season there’s nothing a of can’t fix no waffling about it these cones are to die for wait what is a thimbleberry anyway i don't know but wow

# Now answer/create the following:

1. Top 15 meaningful words

# Create a custom stopword list  
custom\_stopwords <- c(stopwords("en"), "i'm", "im", "m", "s", "t", "ve", "re", "ll", "d", "just") # Removes unwanted short words  
  
# 🔥 Step 1: Pre-clean the text before tokenization  
posts\_filtered\_cleaned <- posts\_filtered\_cleaned %>%  
 mutate(Comment = tolower(Comment), # Convert everything to lowercase  
 Comment = str\_replace\_all(Comment, "\\b(i'm|im)\\b", ""), # Remove "i'm" and "im"  
 Comment = removeWords(Comment, custom\_stopwords), # Remove all stopwords  
 Comment = str\_replace\_all(Comment, "[[:punct:]]", ""), # Remove punctuation  
 Comment = str\_squish(Comment)) # Remove extra spaces  
  
# 🔥 Step 2: Tokenization and Further Cleanup  
top\_words <- posts\_filtered\_cleaned %>%  
 unnest\_tokens(word, Comment) %>%  
 filter(nchar(word) > 1) %>% # Removes single-letter words except "a" and "i"  
 count(word, sort = TRUE) %>%  
 slice\_max(n, n = 15)  
  
# ✅ Step 3: Debugging Check  
print(unique(top\_words$word)) # 🔍 Verify if "m" and "s" are removed

## [1] "like" "life" "one" "cherry" "good" "can" "love" "time"   
## [9] "world" "eat" "every" "pie" "scream" "never" "go"

# ✅ Step 4: View Cleaned Top 15 Words  
top\_words

## word n  
## 1 like 29529  
## 2 life 23025  
## 3 one 20414  
## 4 cherry 20119  
## 5 good 19211  
## 6 can 17191  
## 7 love 13357  
## 8 time 13183  
## 9 world 11455  
## 10 eat 11293  
## 11 every 11012  
## 12 pie 10560  
## 13 scream 9812  
## 14 never 9678  
## 15 go 9604

1. Word cloud

set.seed(123)  
  
# Tokenize comments to words, remove stop words, and count word frequencies  
tokenized\_comments <- posts\_filtered\_cleaned %>%  
 unnest\_tokens(word, Comment) %>%  
 anti\_join(stop\_words) %>%  
 count(word, sort = TRUE)

## Joining with `by = join\_by(word)`

# Generating the word cloud with a custom color palette  
wordcloud(words = tokenized\_comments$word, freq = tokenized\_comments$n, min.freq = 5,  
 max.words = 100, random.order = FALSE, scale = c(3, 0.5),   
 colors = brewer.pal(8, "Dark2"))



1. Who are the top 5 customers by CustomerID who posts the most comments on products?

# Finding the top 5 customers by number of comments with tiebreaker by CustomerID  
top\_customers <- posts\_filtered\_cleaned %>%  
 count(CustomerID, sort = TRUE) %>%  
 arrange(desc(n), CustomerID) %>% # Sorting by n, then by CustomerID for tiebreaking  
 slice\_head(n = 5) # Select the top 5 customers only  
  
top\_customers

## CustomerID n  
## 1 99 38  
## 2 180 38  
## 3 251 38  
## 4 94 37  
## 5 301 37

1. For each these 5 are these positive or negative customers overall considering a measure using the afinn sentiment measure?

afinn <- get\_sentiments("afinn")  
  
# Calculating AFINN Sentiment Score for Top 5 Customers  
sentiment\_top\_customers <- posts\_filtered\_cleaned %>%  
 filter(CustomerID %in% top\_customers$CustomerID) %>%  
 unnest\_tokens(word, Comment) %>%  
 inner\_join(afinn, by = "word") %>%  
 group\_by(CustomerID) %>%  
 summarize(sentiment\_score = sum(value)) %>%  
 arrange(desc(sentiment\_score))  
  
# Calculating Average Star Rating for Top 5 Customers  
average\_rating\_top\_customers <- posts\_filtered\_cleaned %>%  
 filter(CustomerID %in% top\_customers$CustomerID) %>%  
 group\_by(CustomerID) %>%  
 summarize(avg\_rating = mean(Star\_Rating, na.rm = TRUE))  
  
# Combining Sentiment Scores and Ratings, and Classify Sentiment  
combined\_sentiment <- sentiment\_top\_customers %>%  
 inner\_join(average\_rating\_top\_customers, by = "CustomerID") %>%  
 mutate(overall\_sentiment = case\_when(  
 avg\_rating < 3 ~ "negative",  
 avg\_rating >= 3 ~ "positive"  
 ))  
  
# View the overall sentiment for top customers  
combined\_sentiment

## # A tibble: 5 × 4  
## CustomerID sentiment\_score avg\_rating overall\_sentiment  
## <dbl> <dbl> <dbl> <chr>   
## 1 99 1938 4.79 positive   
## 2 180 1786 3.61 positive   
## 3 94 1665 3.97 positive   
## 4 251 1520 3.92 positive   
## 5 301 1517 3.97 positive

1. What Parlor site has the most comments? Are these negative or positive?

# Finding the Parlor Site with the Most Comments  
most\_comments\_parlor <- posts\_filtered\_cleaned %>%  
 count(ParlorLocation, sort = TRUE) %>%  
 slice\_max(n, n = 1)  
  
# Viewing the Parlor Site with the Most Comments  
most\_comments\_parlor

## ParlorLocation n  
## 1 traverse city 5467

# Filtering for the Parlor Site with the Most Comments  
parlor\_with\_most\_comments <- posts\_filtered\_cleaned %>%  
 filter(ParlorLocation == most\_comments\_parlor$ParlorLocation)  
  
# Calculating Sentiment Score Using AFINN  
parlor\_sentiment <- parlor\_with\_most\_comments %>%  
 unnest\_tokens(word, Comment) %>%  
 inner\_join(afinn, by = "word") %>%  
 summarize(sentiment\_score = sum(value))  
  
# Calculating Average Rating for the Parlor Site  
avg\_rating\_parlor <- parlor\_with\_most\_comments %>%  
 summarize(avg\_rating = mean(Star\_Rating, na.rm = TRUE))  
  
# Determining Overall Sentiment for the Parlor Site  
overall\_sentiment <- parlor\_sentiment %>%  
 mutate(  
 ParlorLocation = most\_comments\_parlor$ParlorLocation,  
 num\_comments = most\_comments\_parlor$n,  
 avg\_rating = avg\_rating\_parlor$avg\_rating,  
 overall\_sentiment = case\_when(  
 avg\_rating < 3 ~ "negative",  
 avg\_rating >= 3 ~ "positive"  
 )  
 )  
  
# View the Overall Sentiment for the Parlor with the Most Comments  
overall\_sentiment

## sentiment\_score ParlorLocation num\_comments avg\_rating overall\_sentiment  
## 1 143948 traverse city 5467 4.817816 positive

1. What production site has the most comments? Are these negative or positive?

# Finding the Production Site with the Most Comments  
most\_comments\_production <- posts\_filtered\_cleaned %>%  
 count(ProductionLocation, sort = TRUE) %>%  
 slice\_max(n, n = 1)  
  
# View the Production Site with the Most Comments  
most\_comments\_production

## ProductionLocation n  
## 1 mount pleasant 7952

# Filtering for the Production Site with the Most Comments  
production\_with\_most\_comments <- posts\_filtered\_cleaned %>%  
 filter(ProductionLocation == most\_comments\_production$ProductionLocation)  
  
# Calculating Sentiment Score Using AFINN  
production\_sentiment <- production\_with\_most\_comments %>%  
 unnest\_tokens(word, Comment) %>%  
 inner\_join(afinn, by = "word") %>%  
 summarize(sentiment\_score = sum(value))  
  
# Calculating Average Rating for the Production Site  
avg\_rating\_production <- production\_with\_most\_comments %>%  
 summarize(avg\_rating = mean(Star\_Rating, na.rm = TRUE))  
  
# Determining Overall Sentiment for the Production Site  
overall\_production\_sentiment <- production\_sentiment %>%  
 mutate(overall\_sentiment = case\_when(  
 avg\_rating\_production$avg\_rating < 3 ~ "negative",  
 avg\_rating\_production$avg\_rating >= 3 ~ "positive"  
 ))  
  
# View the Overall Sentiment for the Production Site with the Most Comments  
overall\_production\_sentiment

## sentiment\_score overall\_sentiment  
## 1 201244 positive

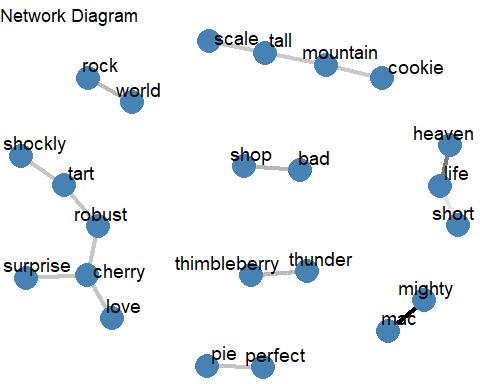
1. What are the top 10 most frequent comments made by customers and how many times did each comment get made?

# Step 1: Count the Frequency of Each Comment  
top\_comments <- posts\_filtered\_cleaned %>%  
 group\_by(Comment) %>%  
 summarize(comment\_count = n()) %>%  
 arrange(desc(comment\_count))  
  
# Step 2: Select the Top 10 Most Frequent Comments Without Ties  
top\_10\_comments <- top\_comments %>%  
 slice\_max(comment\_count, n = 10, with\_ties = FALSE)  
  
# View the Top 10 Most Frequent Comments  
top\_10\_comments

## # A tibble: 10 × 2  
## Comment comment\_count  
## <chr> <int>  
## 1 best prefer sweet potatos really waited line nothing can fix k… 38  
## 2 capturing essence indulgence every head says gym heart says or… 38  
## 3 crazy whoever thought sweet dreams made wow waited ever get se… 38  
## 4 addicted committed relationship strict diet working pretty wel… 37  
## 5 really good prefer sweet potatos committed relationship tub in… 37  
## 6 sweet dreams made can buy happiness can buy kind thing hello h… 37  
## 7 therapy due session life like box chocolates bowl perhaps care… 37  
## 8 wrong make sweet potatos better like cherries one really good … 37  
## 9 cherry top doubt add sprinkles crazy whoever thought strict di… 36  
## 10 chocalate sure get enough strict diet working pretty well neve… 36

1. Create a network diagram based on bigrams

# Tokenize Comments into Bigrams and Remove Stop Words  
bigrams <- posts\_filtered\_cleaned %>%  
 unnest\_tokens(bigram, Comment, token = "ngrams", n = 2) %>%  
 separate(bigram, c("word1", "word2"), sep = " ") %>%  
 filter(!word1 %in% stop\_words$word, !word2 %in% stop\_words$word) %>%  
 filter(word1 != "NA" & word2 != "NA") # Remove any "NA" values  
  
# Count the Frequency of Each Bigram  
bigram\_counts <- bigrams %>%  
 count(word1, word2, sort = TRUE) %>%  
 filter(n > 3) # Lower frequency threshold to only include more frequent bigrams  
  
# Limit to Top 15 Most Frequent Bigrams  
bigram\_top\_15 <- bigram\_counts %>%  
 slice\_max(n, n = 15)  
  
# Create a Graph from the Bigrams  
bigram\_graph <- bigram\_top\_15 %>%  
 graph\_from\_data\_frame()  
  
# Plot the Network Diagram  
set.seed(123)  
ggraph(bigram\_graph, layout = "fr") +  
 geom\_edge\_link(aes(edge\_alpha = n), show.legend = FALSE, edge\_width = 1.5) +  
 geom\_node\_point(color = "steelblue", size = 8) + # Simplify node color and make nodes larger  
 geom\_node\_text(aes(label = name), repel = TRUE, vjust = 1, hjust = 1, size = 5, color = "black") + # Use larger labels for clarity  
 theme\_void() +  
 labs(title = "Network Diagram")



GRADS—- assuming you work for the company/organization for which these product comments have been collected, what can you infer from the data? If the company was asking you if they should take any actions based on customer feedback, what would you tell them and why?

Actions to be taken: Marketing Campaigns: Focus on popular flavors to boost interest and attract new customers.

Customer Testimonials: Use positive customer feedback in promotional material.

Supply Chain Improvements: Ensure the availability of popular products to meet demand. Highlight and promote seasonal or uncommon flavors to generate excitement.