Decision Making with Sentiment Analysis in R

Sanika Parakatil Joseph - 400982266

About the Analysis

Packages Used:

- `tidyverse` – data manipulation and plotting
- `tidytext` – text mining and tokenization
- `lubridate` – date handling
- `ggplot2` – visualisation

Amazon Fine Food Review:

- Product reviews (cookies, coffee, tea, granola)
- Variables: star rating, text, date, product ID
- Goal: quantify review sentiment and relate it to ratings
- Application: support decisions about product quality and marketing

Key Concepts

- **Sentiment analysis** Converts written text into measurable emotional signals by identifying whether language expresses positivity, negativity, or neutrality. This allows organisations to quantify customer feelings instead of relying only on subjective reading.
- **Lexicon-based methods** These approaches classify words by matching them to predefined dictionaries such as the Bing lexicon, which labels words as positive or negative. They are fast, transparent, and easy to interpret, making them ideal for business applications.
- **Sentiment score** Each review receives an overall score by summing the polarity of the words it contains, giving a numerical representation of emotion. Higher scores indicate more positive language, while negative scores highlight dissatisfaction.
- **Decision-making** Analyzing sentiment patterns helps identify products with recurring negative feedback, service gaps, or early warning signs of risk. Managers can use these insights to prioritise improvements, adjust marketing, or intervene before ratings decline.

Step 1: Install & Load Packages

```
#| echo: true
#| message: false
#| warning: false

library(tidyverse)
library(tidytext)
library(lubridate)
library(ggplot2)
library(plotly)
```

Purpose: They provide the core tools for importing data, tokenizing text, applying sentiment lexicons, and creating plots.

Step 2: Load & Prepare Data

```
# A tibble: 6 × 8
     Id ProductId  UserId Score Time       Summary           Text  review_length
  <dbl> <chr>      <chr>  <dbl> <date>     <chr>             <chr>         <int>
1     1 B001E4KFG0 U1         5 0001-01-20 Fantastic cookies Thes…            18
2     2 B001E4KFG0 U2         4 0003-01-20 Good but a bit s… The …            16
3     3 B001E4KFG0 U3         2 0005-01-20 Too dry           I fo…            12
4     4 B008J4RP1U U4         1 0007-01-20 Terrible quality  The …            11
5     5 B008J4RP1U U5         3 0010-01-20 Average snack     Noth…            13
6     6 B000HDK0DC U6         5 0012-01-20 Excellent coffee  Rich…            12
```

Purpose: We use a small Amazon Fine Food–style dataset.
Each row is a review with a star rating (Score) and review text (Text).
We convert the time variable to a Date and compute review length (number of words).

Step 3: Tokenize Text & Join Sentiment Lexicon

```
bing_lexicon <- get_sentiments("bing")

tokens <- reviews %>%
  unnest_tokens(word, Text)

sentiment_words <- tokens %>%
  inner_join(bing_lexicon, by = "word")

head(sentiment_words)
```

```
# A tibble: 6 × 9
     Id ProductId  UserId Score Time       Summary review_length word  sentiment
  <dbl> <chr>      <chr>  <dbl> <date>     <chr>           <int> <chr> <chr>
1     1 B001E4KFG0 U1         5 0001-01-20 Fantas…            18 amaz… positive
2     1 B001E4KFG0 U1         5 0001-01-20 Fantas…            18 fresh positive
3     1 B001E4KFG0 U1         5 0001-01-20 Fantas…            18 happy positive
4     2 B001E4KFG0 U2         4 0003-01-20 Good b…            16 good  positive
5     2 B001E4KFG0 U2         4 0003-01-20 Good b…            16 sweet positive
6     3 B001E4KFG0 U3         2 0005-01-20 Too dry            12 disa… negative
```

Purpose:unnest_tokens() splits each review into individual words.
The Bing lexicon classifies words as positive or negative. The join keeps only words that carry sentiment.

Step 4: Compute Sentiment Scores

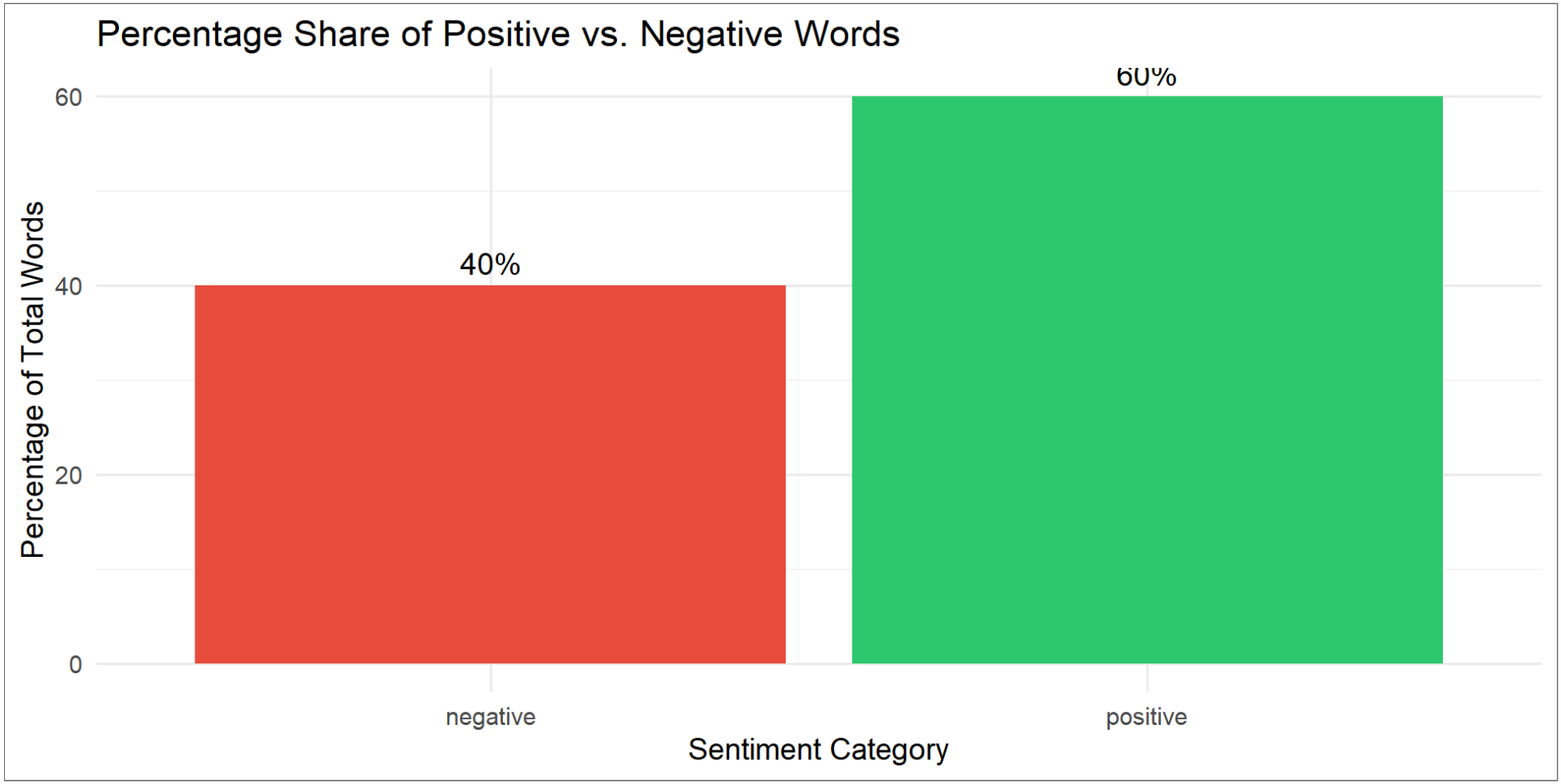
```r
review_sentiment <- sentiment_words %>%
  mutate(score = if_else(sentiment == "positive", 1, -1)) %>%
  group_by(Id) %>%
  summarise(sentiment_score = sum(score), .groups = "drop")

reviews_scored <- reviews %>%
  left_join(review_sentiment, by = "Id") %>%
  mutate(sentiment_score = replace_na(sentiment_score, 0))

reviews_scored
```

```
# A tibble: 12 × 9
      Id ProductId  UserId Score Time       Summary          Text  review_length
   <dbl> <chr>      <chr>  <dbl> <date>     <chr>            <chr>         <int>
 1     1 B001E4KFG0 U1         5 0001-01-20 Fantastic cooki… Thes…            18
 2     2 B001E4KFG0 U2         4 0003-01-20 Good but a bit … The …            16
 3     3 B001E4KFG0 U3         2 0005-01-20 Too dry          I fo…            12
 4     4 B008J4RP1U U4         1 0007-01-20 Terrible quality The …            11
 5     5 B008J4RP1U U5         3 0010-01-20 Average snack    Noth…            13
 6     6 B000HDK0DC U6         5 0012-01-20 Excellent coffee Rich…            12
 7     7 B000HDK0DC U7         4 0014-01-20 Good but expens… The …            14
 8     8 B000HDK0DC U8         2 0016-01-20 Bitter and burnt The …            13
 9     9 B000SQNQIQ U9         5 0018-01-20 Perfect tea      This…            14
10    10 B000SQNQIQ U10        3 0019-01-20 Decent but weak  The …            15
11    11 B000SQNQIQ U11        1 0021-01-20 Worst tea ever   This…            19
12    12 B002HQ0Z1U U12        4 0023-01-20 Tasty granola    The …            15
# ℹ 1 more variable: sentiment_score <dbl>
```

Key Point: Each positive word contributes +1; each negative word contributes −1.

Scores are summed at the review level and Reviews with no sentiment words receive a score of 0.

Step 5: Visualising Sentiment (Bar Plot)

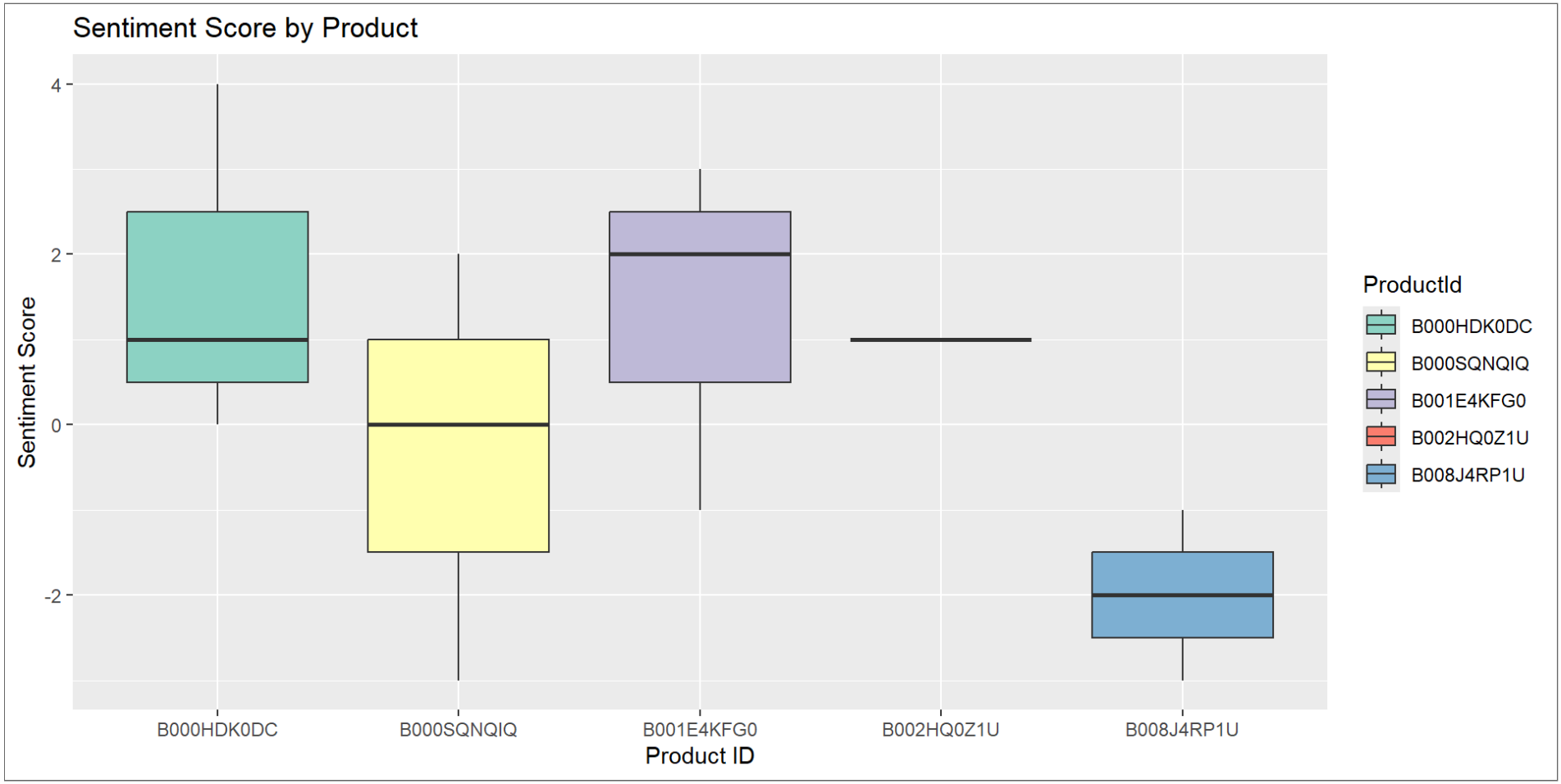


Key Point: This bar chart shows whether positive or negative words dominate the sample. Managers can quickly assess whether feedback is generally favourable or problematic

Step 6: Rating vs. Sentiment Score (Scatter Plot)

Observation: Reviews with higher star ratings tend to have higher sentiment scores.
If many low-score reviews have strongly negative sentiment, the product may require improvement.

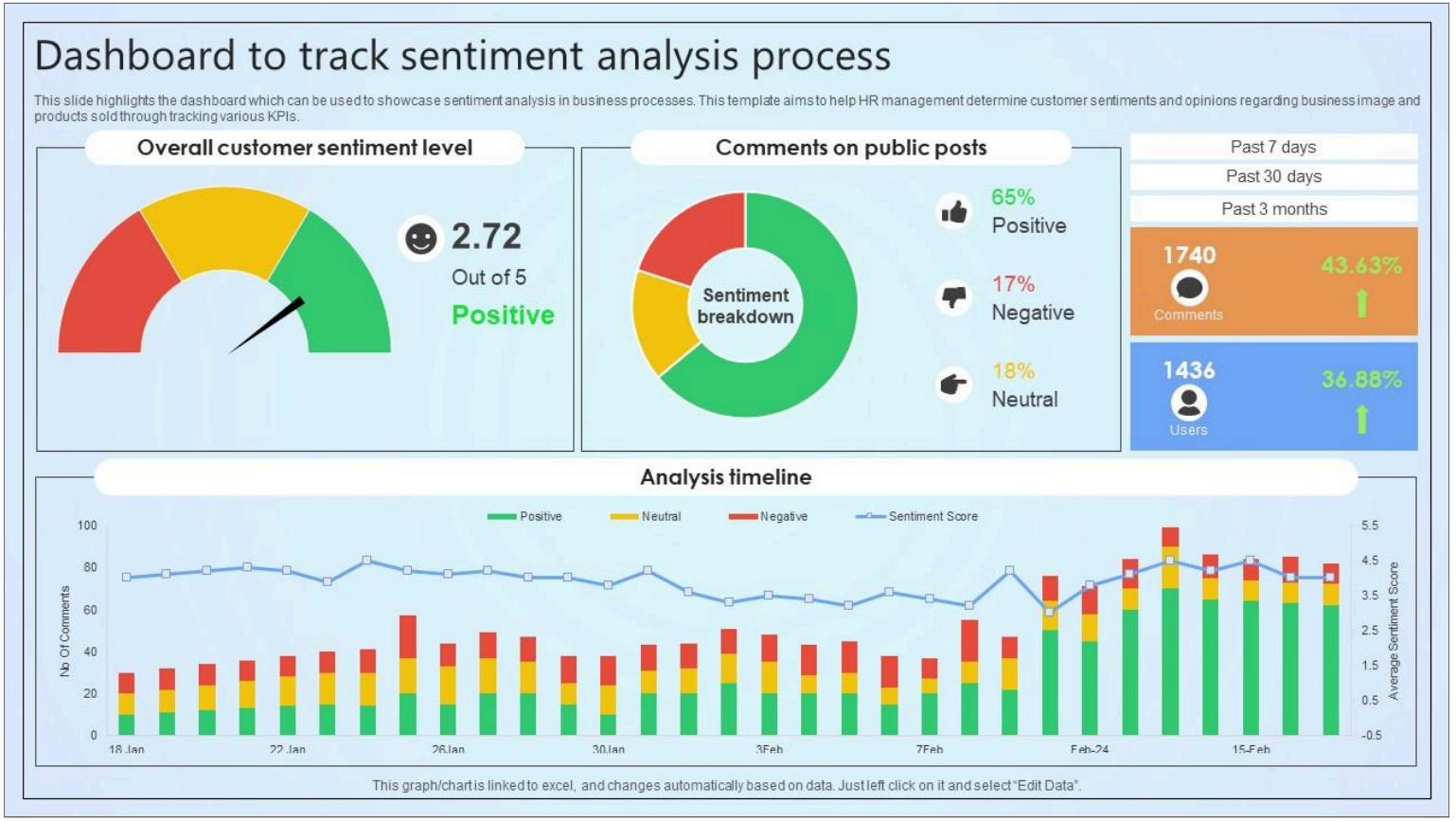
Sentiment by Product (Boxplot)



Observation: This plot compares sentiment across different products. Products with lower median scores or many negative outliers may need closer attention

Business Applications

- **Product quality management**: Identify items with consistently negative sentiment and prioritise quality checks.
- **Customer service**: Detect themes in negative reviews and design targeted service interventions.
- **Marketing**: Emphasise features associated with strongly positive sentiment in campaigns.
- **Risk monitoring**: Track sentiment over time to detect early warning signals before ratings drop.



)

Exercise: Try With Another Dataset

- Use the same code structure with a different text dataset (e.g., your own product reviews).
- Replace `amazon_fine_food_review.csv` with your file.
- Re-run tokenisation, lexicon matching, scoring, and visualisation.
- Compare whether the relationship between ratings and sentiment is similar.

Conclusion

- Sentiment analysis reveals emotional patterns in customer reviews.
- R makes it easy to score, visualize, and interpret sentiment at scale.
- Insights help managers spot issues early and make better product decisions.
- A simple method with powerful impact on decision-making.

Thank You!

Questions?

Speaker notes