HSS 371 Data Project Reproducibility materials

2024-05-10

Gemini Function

The `gemini` function was created in order to utilize Google's Gemini API for text generation and sentiment analysis. It takes several parameters, including `prompts`, `temperature`, `max_output_tokens`, `api_key`, and `model`. Here's a breakdown of each parameter:

- `prompts`: A vector containing the text prompts for which sentiment analysis is to be performed.
- `temperature`: A parameter controlling the randomness of the text generation process. Higher values result in more random outputs.
- `max_output_tokens`: The maximum number of tokens in the generated text.
- `api_key`: The API key required for accessing the Gemini API. If not provided, the function prompts the user to input the API key.
- `model`: The name of the model to be used for text generation. Default is "gemini-pro".

The function sends requests to the Gemini API, processes the responses, and returns the generated text along with sentiment analysis results. More information of the source code can be found here (https://github.com/wilkflow/qssProj/blob/main/317_Gemini_LLM.R)

Gemini Output Cleaning and Output Analysis (Stacked Barplot)

Figure 1 *Stacked Crowdworker Sentiment Distribution (Crowdworkers vs Gemini Classification)*

```
data <- read.csv("text_emotion_with_gemini.csv")
data = na.omit(data)

text_emotion <- read.csv("text_emotion_with_gemini.csv")
head(text_emotion) #optional: check the beginning of the dataset
```

```
##     tweet_id sentiment        author
## 1 1956967341     empty    xoshayzers
## 2 1956969456   neutral    feinyheiny
## 3 1956971981     worry andreagauster
## 4 1956974706      hate   MavrickAces
## 5 1956977084 happiness      ktierson
## 6 1956979894   neutral  lookitsholly
##                                                                                 conten
t
## 1 @tiffanylue i know  i was listenin to bad habit earlier and i started freakin at his part =
[
## 2                                                                              cant fall aslee
p
## 3       @raaaaaaek oh too bad! I hope it gets better. I've been having sleep issues lately to
o
## 4         It is so annoying when she starts typing on her computer in the middle of the nigh
t!
## 5                      mmm much better day... so far! it's still quite early. last day of #ud
s
## 6                          Chocolate milk is so much better through a straw. I lack said stra
w
##   gemini
## 1     12
## 2     11
## 3     10
## 4      1
## 5      6
## 6      2
```

```
#remove all NA's to only analyze Gemini's performance
text_emotion_only_gemini <- na.omit(text_emotion)
head(text_emotion_only_gemini) #optional: verify that the dataset was cleaned
```

```
##     tweet_id sentiment        author
## 1 1956967341     empty    xoshayzers
## 2 1956969456   neutral    feinyheiny
## 3 1956971981     worry andreagauster
## 4 1956974706      hate   MavrickAces
## 5 1956977084 happiness      ktierson
## 6 1956979894   neutral  lookitsholly
##                                                                                 conten
t
## 1 @tiffanylue i know  i was listenin to bad habit earlier and i started freakin at his part =
[
## 2                                                                          cant fall aslee
p
## 3       @raaaaaaek oh too bad! I hope it gets better. I've been having sleep issues lately to
o
## 4         It is so annoying when she starts typing on her computer in the middle of the nigh
t!
## 5                      mmm much better day... so far! it's still quite early. last day of #ud
s
## 6                          Chocolate milk is so much better through a straw. I lack said stra
w
##   gemini
## 1     12
## 2     11
## 3     10
## 4      1
## 5      6
## 6      2
```

```
# Define a function to map numeric values to sentiment words
map_sentiment <- function(value) {
  sentiment_map <- c("anger", "boredom", "empty", "enthusiasm", "fun",
                     "happiness", "hate", "love", "neutral", "relief",
                     "sadness", "surprise", "worry")
  return(sentiment_map[value])
}
data$gemini <- sapply(data$gemini, map_sentiment)

# Convert the numeric values in the gemini column to sentiment words
text_emotion_only_gemini$gemini <- sapply(text_emotion_only_gemini$gemini, map_sentiment)

#print(text_emotion_only_gemini)
# Optional: View the updated dataset
# Load required library for plotting
# Convert the sentiment_distribution data into a long format
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.2.3
```

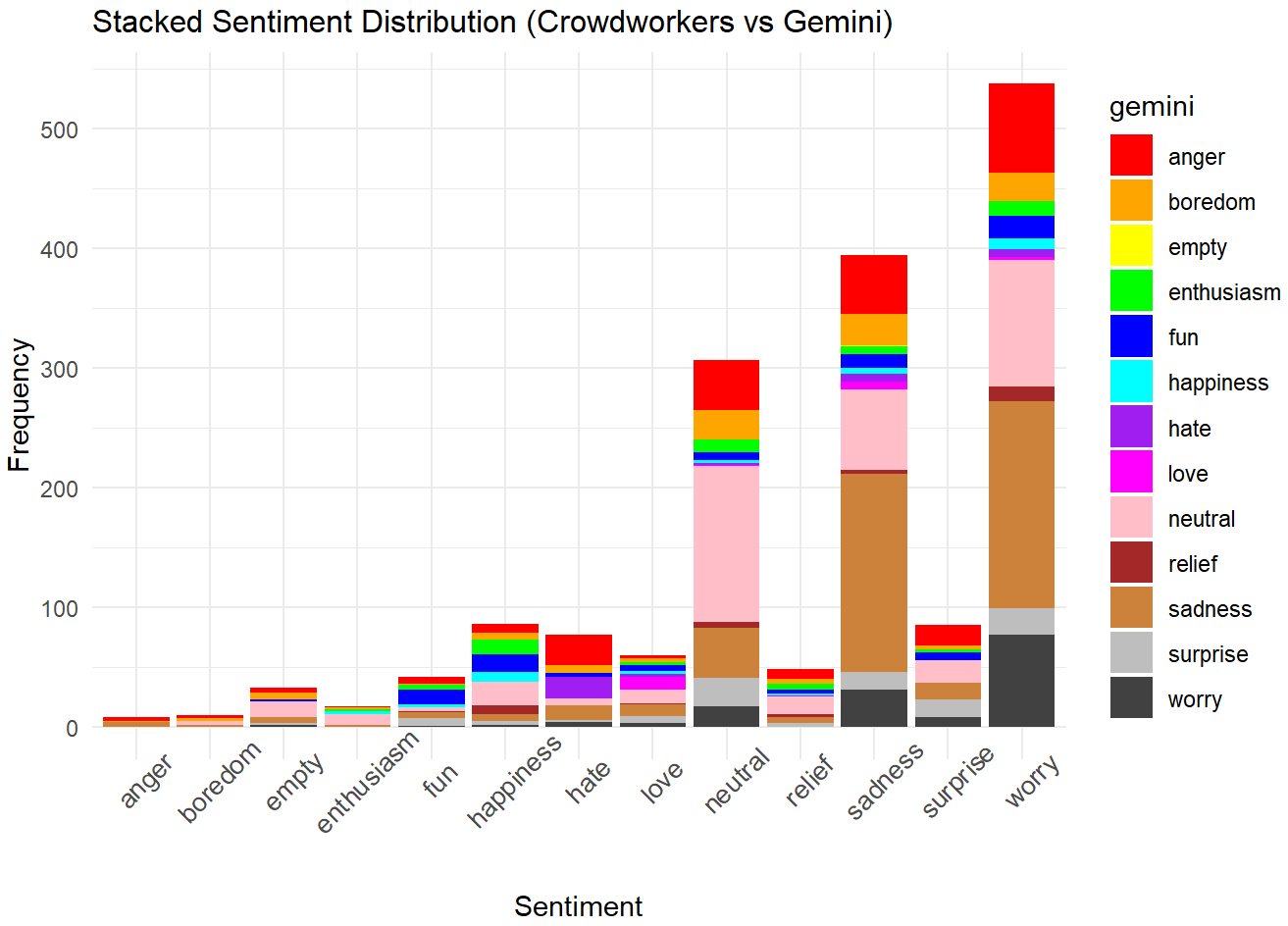
```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
# Convert the rownames of sentiment_distribution to a new column
sentiment_distribution <- table(text_emotion_only_gemini$sentiment, text_emotion_only_gemini$gem
ini)
sentiment_distribution <- as.data.frame.matrix(sentiment_distribution)
sentiment_distribution$sentiment <- rownames(sentiment_distribution)

# Define custom colors for the bar plot
# Define breaks and labels for y-axis
breaks <- seq(0, 600, by = 100)
labels <- seq(0, 600, by = 100)
custom_colors <- c("red", "orange", "yellow", "green", "blue", "cyan", "purple", "magenta", "pin
k", "brown", "tan3", "gray", "gray26")
sentiment_long <- tidyr::pivot_longer(sentiment_distribution, cols = -sentiment, names_to = "gem
ini", values_to = "count")

# Create a stacked bar chart of sentiment distribution with custom colors
ggplot(sentiment_long, aes(x = sentiment, y = count, fill = gemini)) +
  geom_bar(stat = "identity",) +
  labs(title = "Stacked Sentiment Distribution (Crowdworkers vs Gemini)",
       x = "Sentiment",
       y = "Frequency") +
  scale_fill_manual(values = custom_colors) +  # Apply custom colors
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 0.5, size = 10), plot.title = element_tex
t(size = 12, hjust = 0)) +
  scale_y_continuous(breaks = breaks, labels = labels) # Text adjustment for the x and y axis re
spectively
```



##Confusion Matrix and Heatmap: Crowdworkers vs Gemini **Figure 2** *Confusion Matrix: Accuracy Between Gemini and Crowdworker Classification*

```
# Load necessary library
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.3
```

```
## Loading required package: lattice
```

```
 # Calculate confusion matrix
 c = confusionMatrix(as.factor(data$sentiment), as.factor(data$gemini), positive = NULL, dnn = c
("Prediction", "Gemini"))
 # Convert confusion matrix to data frame
plt <- as.data.frame(c$table)
ggplot(plt, aes(Prediction, rev(Gemini), fill = Freq)) +
  geom_tile() +
  geom_text(aes(label = Freq)) +
  scale_fill_gradient(low = "white", high = "#009194") +
  labs(x = "Crowdworker Classification", y = "Gemini") +
  scale_x_discrete(labels = c("anger", "boredom", "empty", "enthusiasm", "fun",
                              "happiness", "hate", "love", "neutral", "relief",
                              "sadness", "surprise", "worry")) +
  scale_y_discrete(labels = rev(c("anger", "boredom", "empty", "enthusiasm", "fun",
                                  "happiness", "hate", "love", "neutral", "relief",
                                  "sadness", "surprise", "worry"))) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

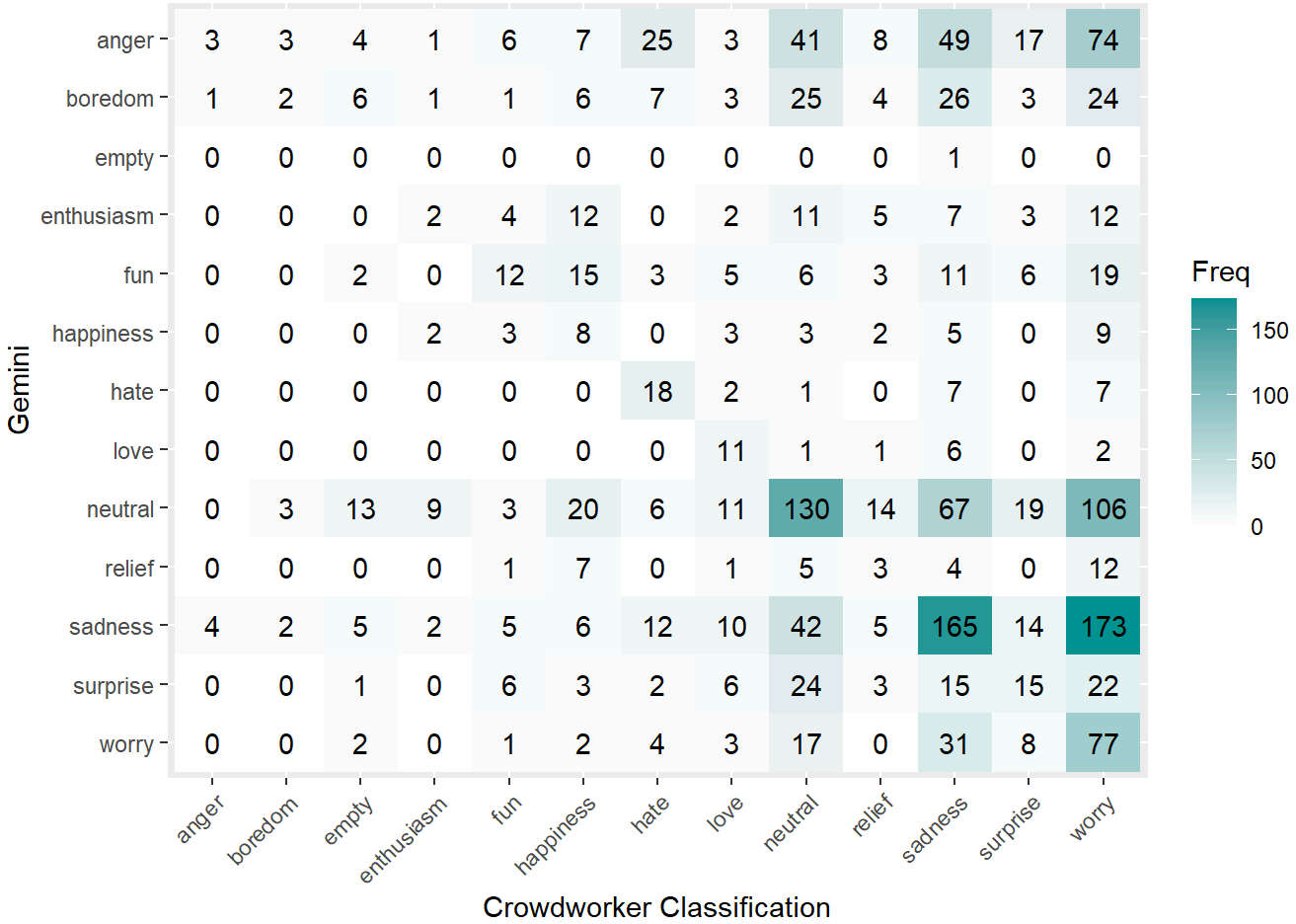


Figure 2.5 *Confusion matrix output*

```
print(c[3])
```

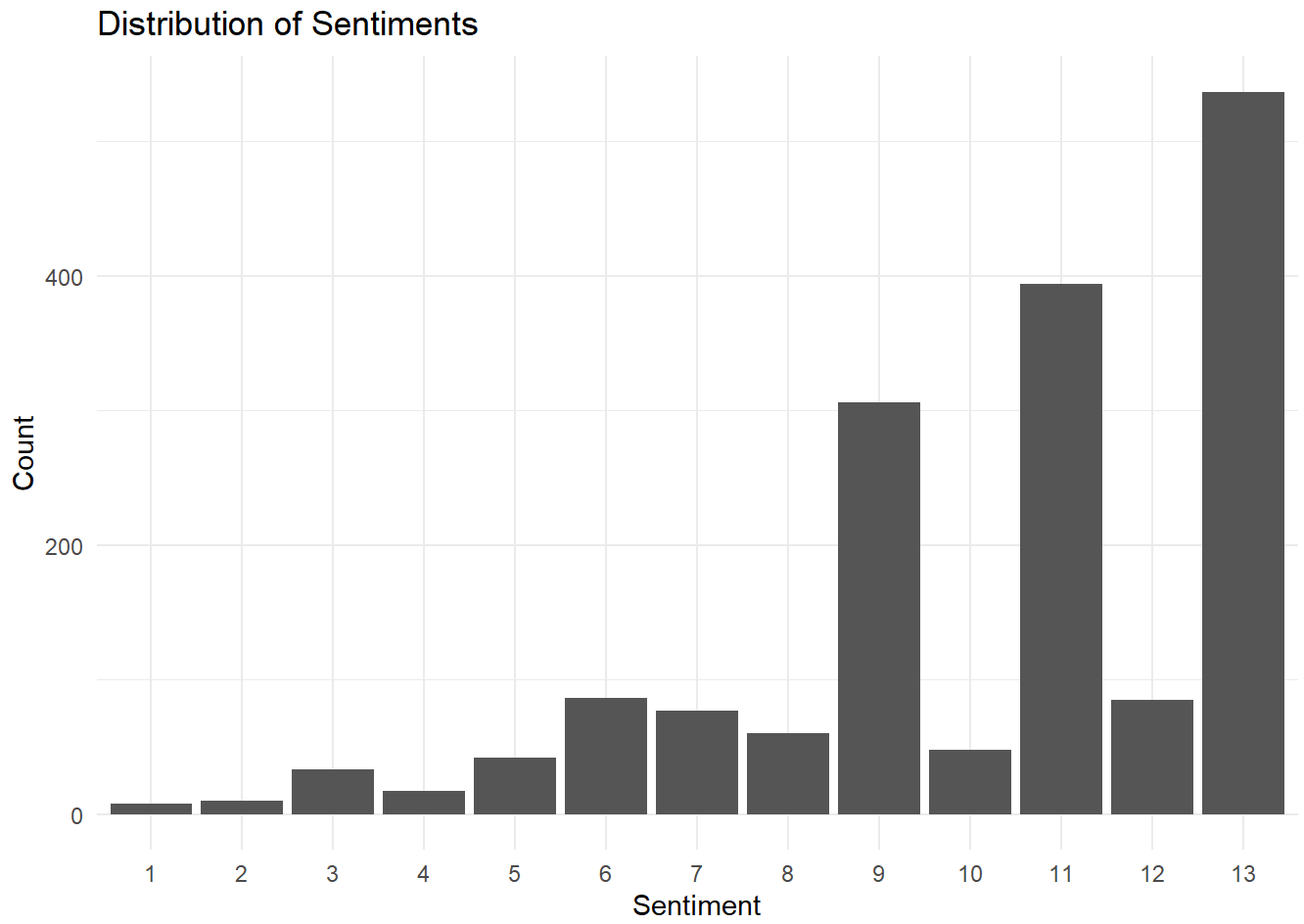
```
## $overall
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.2618908      0.1437429      0.2411405      0.2834647      0.2613036
## AccuracyPValue  McnemarPValue
##      0.4872510            NaN
```

Figure 3 *Crowdworker Sentiment Classification: Distribution of Sentiments*

```
sentiment_mapping <- c('anger' = 1, 'boredom' = 2, 'empty' = 3, 'enthusiasm' = 4,
                       'fun' = 5, 'happiness' = 6, 'hate' = 7, 'love' = 8,
                       'neutral' = 9, 'relief' = 10, 'sadness' = 11, 'surprise' = 12,
                       'worry' = 13)
data$sentiment <- as.integer(factor(data$sentiment, levels = names(sentiment_mapping), labels =
sentiment_mapping))

sentiment_counts <- table(data$sentiment)

ggplot(data, aes(x=factor(sentiment, levels = names(sentiment_counts)))) +
  geom_bar() +
  labs(x = "Sentiment", y = "Count", title = "Distribution of Sentiments") +
  theme_minimal()
```



Distribution Gemini API Sentiments

Figure 4 *Gemini Sentiment Classification: Distribution of Sentiments*

```
head(data)
```

```
##     tweet_id sentiment        author
## 1 1956967341         3    xoshayzers
## 2 1956969456         9    feinyheiny
## 3 1956971981        13 andreagauster
## 4 1956974706         7   MavrickAces
## 5 1956977084         6      ktierson
## 6 1956979894         9  lookitsholly
##                                                                                 conten
t
## 1 @tiffanylue i know  i was listenin to bad habit earlier and i started freakin at his part =
[
## 2                                                                             cant fall aslee
p
## 3       @raaaaaaek oh too bad! I hope it gets better. I've been having sleep issues lately to
o
## 4         It is so annoying when she starts typing on her computer in the middle of the nigh
t!
## 5                       mmm much better day... so far! it's still quite early. last day of #ud
s
## 6                           Chocolate milk is so much better through a straw. I lack said stra
w
##      gemini
## 1  surprise
## 2   sadness
## 3    relief
## 4     anger
## 5 happiness
## 6   boredom
```

```
str(data)
```

```
## 'data.frame':    1703 obs. of  5 variables:
##  $ tweet_id : int  1956967341 1956969456 1956971981 1956974706 1956977084 1956979894 19569824
49 1956983931 1956985764 1956988145 ...
##  $ sentiment: int  3 9 13 7 6 9 13 9 11 9 ...
##  $ author   : chr  "xoshayzers" "feinyheiny" "andreagauster" "MavrickAces" ...
##  $ content  : chr  "@tiffanylue i know  i was listenin to bad habit earlier and i started fre
akin at his part =[" "cant fall asleep" "@raaaaaaek oh too bad! I hope it gets better. I've been
having sleep issues lately too" "It is so annoying when she starts typing on her computer in the
middle of the night!" ...
##  $ gemini   : chr  "surprise" "sadness" "relief" "anger" ...
##  - attr(*, "na.action")= 'omit' Named int [1:54] 43 65 83 143 153 165 193 200 223 297 ...
##   ..- attr(*, "names")= chr [1:54] "43" "65" "83" "143" ...
```

```
gemini_counts <- table(data$gemini)

ggplot(data, aes(x=factor(gemini, levels = names(gemini_counts)))) +
  geom_bar(fill = "steelblue") +
  labs(x = "Gemini Sentiment", y = "Count", title = "Distribution of Gemini API Sentiments") +
  theme_minimal()
```

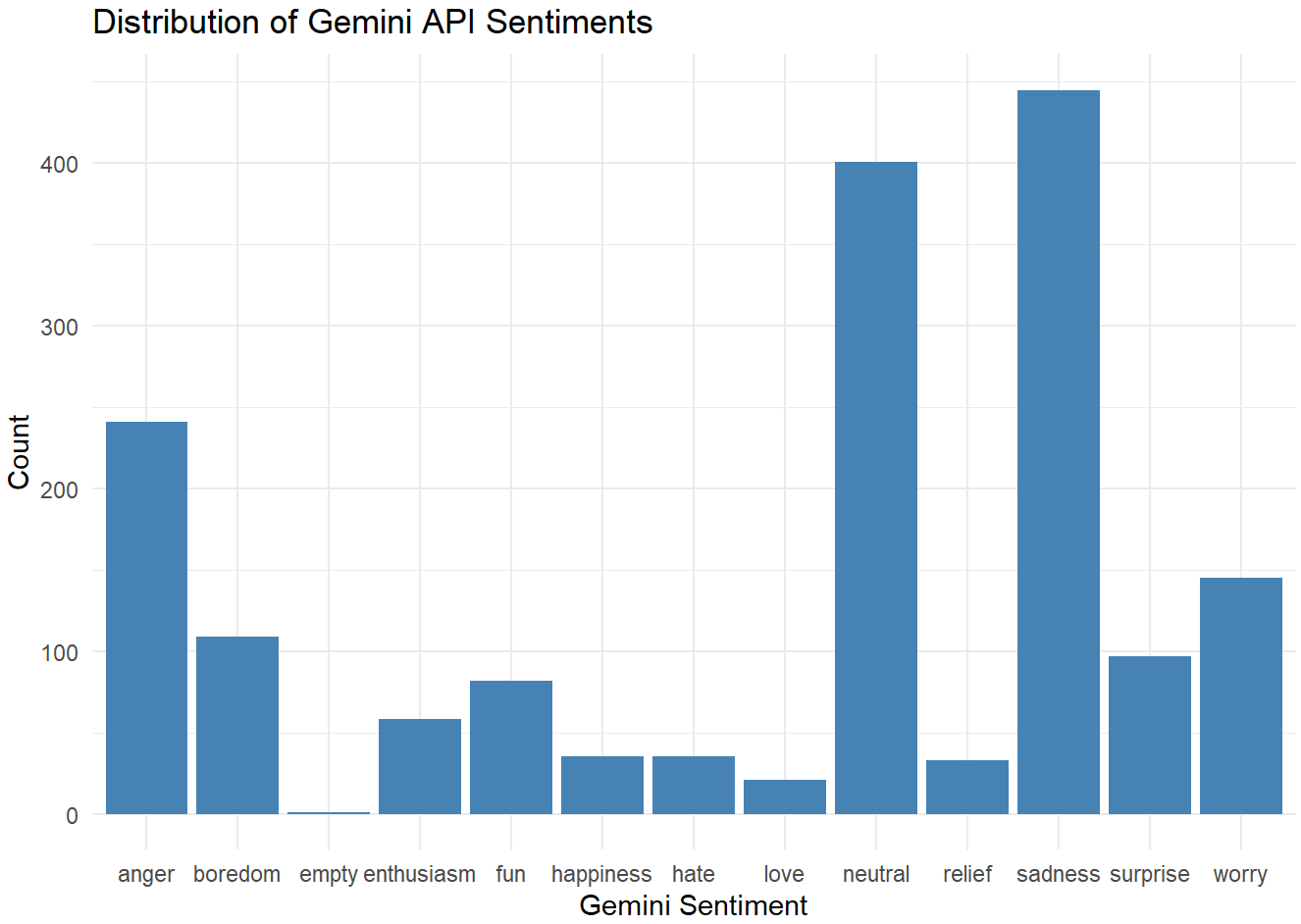


Figure 5 *Gemini vs Crowdworker Sentiment Classification: Distribution of Sentiments*

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

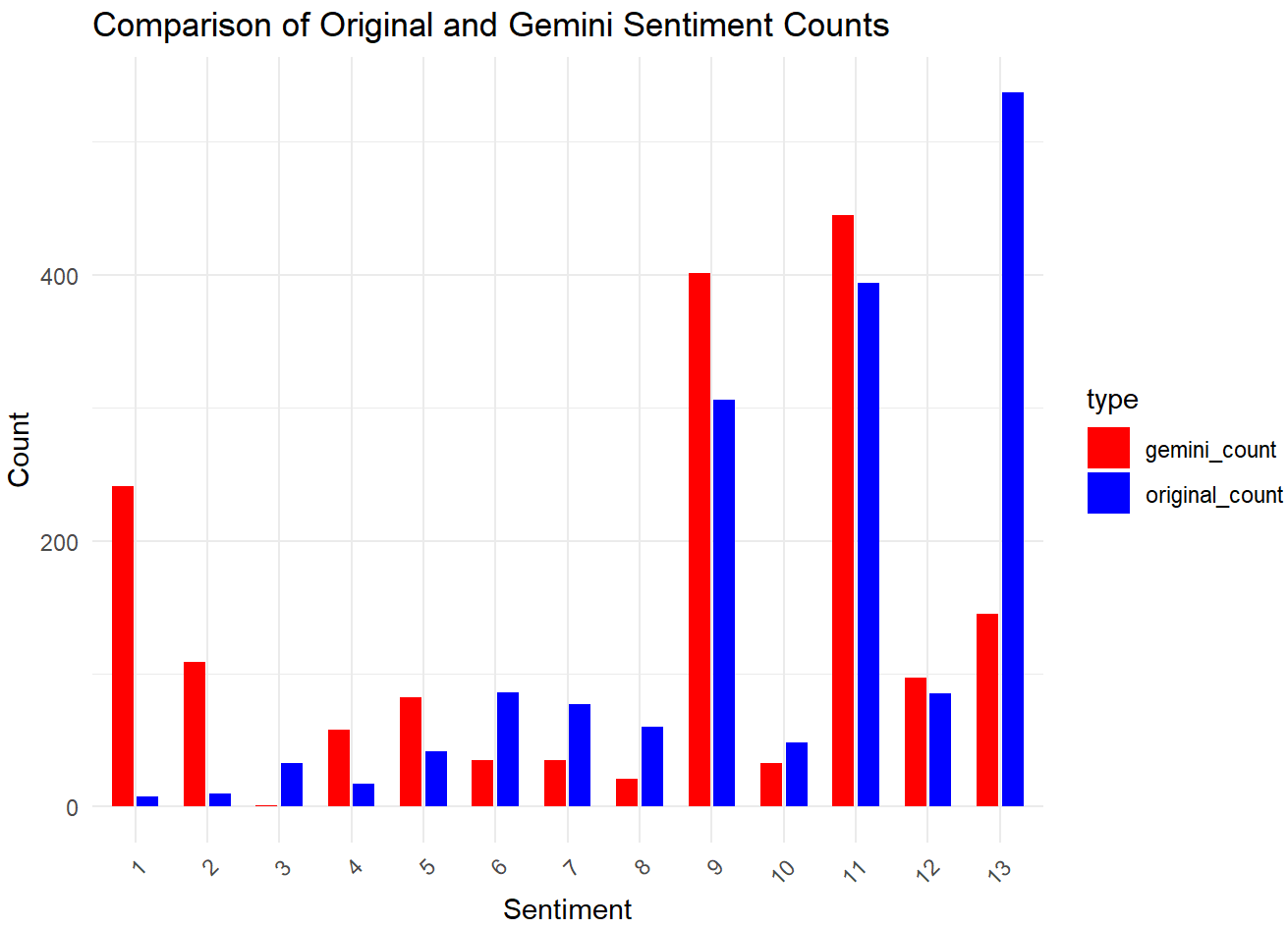
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data <- read.csv("text_emotion_with_gemini.csv")
data = na.omit(data)
sentiment_mapping <- c('anger' = 1, 'boredom' = 2, 'empty' = 3, 'enthusiasm' = 4,
                       'fun' = 5, 'happiness' = 6, 'hate' = 7, 'love' = 8,
                       'neutral' = 9, 'relief' = 10, 'sadness' = 11, 'surprise' = 12,
                       'worry' = 13)
data$sentiment <- as.integer(factor(data$sentiment, levels = names(sentiment_mapping), labels =
sentiment_mapping))
original_counts <- data %>%
  count(sentiment, name = "original_count")
gemini_counts <- data %>%
  count(gemini, name = "gemini_count")
levels_sentiment <- sort(as.numeric(unique(c(as.character(original_counts$sentiment), as.charact
er(gemini_counts$gemini)))))
levels_sentiment <- as.character(levels_sentiment)
levels_sentiment[is.na(levels_sentiment)] <- "NA"

original_counts$sentiment <- factor(original_counts$sentiment, levels = levels_sentiment)
gemini_counts$gemini <- factor(gemini_counts$gemini, levels = levels_sentiment)

combined_counts <- full_join(original_counts, gemini_counts, by = c("sentiment" = "gemini"))
plot_data <- tidyr::pivot_longer(combined_counts, cols = c("original_count", "gemini_count"),
                                 names_to = "type", values_to = "count")
plot_data$count[is.na(plot_data$count)] <- 0

ggplot(plot_data, aes(x = sentiment, y = count, fill = type)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.7), width = 0.6) +
  scale_fill_manual(values = c("original_count" = "blue", "gemini_count" = "red")) +
  labs(x = "Sentiment", y = "Count", title = "Comparison of Original and Gemini Sentiment Count
s") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



##Chi-Squared Test

```
data$gemini <- factor(data$gemini)
data$sentiment <- as.factor(data$sentiment)
table_data <- table(data$gemini, data$sentiment)
chi_squared_result <- chisq.test(table_data)
```

```
## Warning in chisq.test(table_data): Chi-squared approximation may be incorrect
```

```
chi_squared_result
```

```
##
##  Pearson's Chi-squared test
##
## data:  table_data
## X-squared = 870.63, df = 144, p-value < 2.2e-16
```