

✓ 1. Introduction and hypothesis

Introduction Food plays a central role in shaping and reflecting cultural, regional, and social identities. By examining the language used to describe food in historical texts, we can uncover patterns that reveal broader societal norms, regional traditions, and gender roles. This study analyzes a corpus of U.S. fiction texts to investigate the representation of food-related terms across different regions, genders, and time periods. Specifically, we explore how food vocabularies diverge geographically, differ between genders, and highlight unique cultural practices associated with the North and South.

The KL Divergence method is employed to measure the divergence in food-related language across regions and genders, while the Fighting Words analysis identifies the most distinctive food terms used by each group. This approach allows us to pinpoint significant differences in how food is represented and understood in various sociocultural contexts.

Hypotheses

Regional Variation: Food-related terms will vary significantly across geographic regions due to differing cultural, agricultural, and culinary practices. For example, Southern texts will prioritize staple and protein-rich foods like "corn" and "bacon," while Northern texts will emphasize terms like "bread" and "flour."

Gender Differences: Gendered representations of food will reflect traditional gender roles in historical contexts. Female-associated texts will focus on desserts, beverages, and meal preparation (e.g., "tea," "cake," "porridge"), whereas male-associated texts will emphasize staple foods and meats (e.g., "bacon," "turkey").

North vs. South Divergence: Texts from the North and South will exhibit distinct food vocabularies, with Northern texts featuring terms related to hearty meals and meats, while Southern texts will reflect simpler, agrarian staples like "milk" and "bread."

Research Questions

1. How do food-related terms differ across geographic regions in U.S. fiction texts?

Using KL Divergence, we investigate the extent to which food vocabularies vary between regions and highlight distinctive food terms that characterize specific geographic locations, particularly focusing on the North versus the South.

2. What are the gender-based differences in food-related language in historical texts?

Through the Fighting Words analysis and log-odds ratio calculations, we identify the food terms that are significantly more associated with texts labeled as "female" versus "male." How do these differences reflect gendered cultural roles or food perceptions?

3. Which food terms are the most distinctive for different groups (regions, genders, and time periods), and how do these words reflect cultural or social differences?

We aim to highlight the most unique and "fighting" food-related words for each comparison group (e.g., regional or gender-based) and interpret their cultural significance through visualizations.

Methods

1. Data Preparation

Dataset: The dataset comprises fictional texts sourced from various regions, time periods, and genders in U.S. literature. Each text was associated with metadata, including information such as `pub_place` (publication location), `pub_date`, `gender`, and `time_period`.

Text Processing:

Text Loading: Text files were loaded into Python, and the contents were processed to extract meaningful information. Non-UTF-8 characters were ignored to ensure clean text data.

Tokenization: Sentences were split using regular expressions based on punctuation (e.g., . or ?) to isolate individual sentences.

Food Terms Filtering: A predefined lexicon of food-related terms (e.g., bread, soup, meat, cake) was applied to identify and retain sentences containing food-related language.

Text Cleaning: All text was converted to lowercase to ensure uniformity during analysis.

Metadata Integration:

Extracted food-related text was merged with the metadata file to align textual data with associated attributes such as `pub_place`, `gender`, and `time_period`. Token counts for food terms were calculated for each text using `Counter` from the `collections` library.

2. Linguistic Analysis

- **Token Probabilities:** Food Term Frequency Calculation: For each text, token frequencies of food-related terms were calculated using `Counter`.
- **KL Divergence:** Purpose:

KL Divergence was used to quantify the differences in food-related word distributions between groups. A higher KL Divergence score indicates that two distributions are more distinct.

Implementation:

Food token counts for each group were normalized into probability distributions. Pairwise comparisons were performed between groups (e.g., regions, genders, or time periods).

Visualizations:

Heatmaps were generated to display the magnitude of KL Divergence scores, highlighting the most divergent regions, genders, or time periods.

- **Fighting Words:** Purpose:

Fighting Words analysis was used to identify the most distinctive food-related terms between two comparison groups based on the log-odds ratio.

Implementation:

Word frequencies were calculated separately for each group (e.g., Male vs. Female or North vs. South). Log-odds ratios were computed to determine which food terms were significantly more likely to occur in one group compared to another.

Filtering: Non-food terms were excluded to focus solely on food-related language.

Visualizations:

Bar charts were created to visualize the top food-related terms with the highest log-odds ratios, providing insights into distinctive word usage patterns.

- **Comparative Analysis Across Groups:**

Region-Based Comparisons: Regions were grouped into North and South categories, and KL Divergence scores were computed to identify significant differences in food vocabularies.

Gender-Based Comparisons: Food term distributions were compared between texts labeled as "Female" and "Male" using Fighting Words analysis.

Temporal Comparisons: Time periods were analyzed using KL Divergence to evaluate how food-related language evolved over time.

Visualization and Interpretation

-KL Divergence Heatmaps: Displayed KL Divergence scores for regional and temporal comparisons, providing an overview of distributional differences.

-Fighting Words Word Clouds: Generate word clouds to visualize food terms that are statistically overrepresented (Fighting Words) in different regions, gender groups, and ethnicities. Use color coding to distinguish between groups.

-Bar Charts: Illustrated the top distinctive food terms (based on log-odds ratios) for gender-based and region-based comparisons.

-Top Words Plots: Highlighted the most prominent food-related terms specific to each comparison group, offering insights into cultural and social differences.

-Geographic Maps: Represent food-related linguistic and thematic trends spatially across the U.S. Use `pub_place` or `state_main` metadata to map the frequency and diversity of food-related terms by region, with pre- and post-Civil War comparisons.

Expected Insights

1. Regional Divergence:

- Southern texts may emphasize agrarian and communal food practices, such as mentions of "corn," "feasts," and "roasts," reflecting rural lifestyles and traditions.
- Northern texts may highlight urban dining, industrial food production, and scarcity during times of war, with terms like "bread," "factory," or "ration."
- Pre- and post-Civil War analysis may reveal a decline in references to abundance in Southern literature, replaced by themes of scarcity and survival.

2. Ethnic Divergence:

- African American authors may highlight food as a symbol of resilience, cultural memory, and survival, especially post-Civil War, with mentions of "cornbread," "meal," and "fields."
- Indigenous authors may include references to traditional food practices and connections to nature, with terms like "fish," "hunt," and "harvest."
- White authors may reflect dominant cultural narratives, such as wealth, feasts, or scarcity during wartime, with significant shifts in tone across time.

3. Gendered Patterns:

- Female authors may focus on domestic meals, kitchens, and nurturing roles, with frequent mentions of "tea," "kitchen," "sugar," and "baking."
- Male authors may emphasize hunting, feasts, and public meals, reflecting traditional gender norms, with words like "hunt," "meat," and "feast" being prominent.
- Gendered differences in food-related sentiment may show female authors portraying food in warmer, more familial contexts, while male authors depict it as part of adventure or survival.

4. Historical Trends:

- Food-related themes are likely to evolve before and after the Civil War, with pre-war texts reflecting abundance and celebration, while post-war texts shift to scarcity, survival, and loss.
- The prominence of food in literature may increase during times of hardship, such as wartime periods, as a marker of societal struggles.

5. Linguistic Divergence:

- KL Divergence will quantify the extent of regional and cultural differences in food-related vocabulary, showing clear linguistic distinctions between North and South, as well as between ethnic groups.
- Fighting Words will highlight food terms unique to specific groups, serving as cultural markers of identity and lifestyle.

Significance

This study provides a unique lens for understanding how food serves as a cultural and social marker in 19th-century American fiction. By analyzing food-related language and themes across regions, gender, and ethnicity, as well as tracking changes before and after the Civil War, this project contributes to both literary studies and cultural history in the following ways:

1. **Revealing Cultural Identity:** Food-related language reflects cultural practices, traditions, and values. Examining regional and ethnic differences highlights how food connects to identity, survival, and heritage, particularly for underrepresented groups like African American and Indigenous authors.
2. **Gender and Social Norms:** By identifying gendered differences in food representations, this study uncovers how authors use food to reinforce or challenge societal roles, such as women's association with domestic spaces and men's focus on hunting and feasting.
3. **Historical Context and Change:** Tracking food-related themes over time offers insight into how historical events, such as the Civil War, impacted everyday life and literary expression. Food references shift from symbols of abundance to markers of hardship, revealing the societal struggles of the era.

Field definitions and distributional stats

Most of the metadata fields are self-explanatory, but here are some details. Note that not every field in the metadata is described below.


- `source_id`: This is the name of the file corresponding to the volume. You can use it to match metadata records to full-text documents. Note that the corpus includes a nontrivial number of multivolume works. These volumes have `source_id`s like `eaf086v1` or `Wright2-1720v2`.
- `gender`: Author gender. `M`, `F`, or `NaN` (= unknown).
- `gender_guess`: Was the author gender assignment determined by biographical research (`0`) or by guessing on the basis of the author's name (`1`)?
- `ethnicity`: Author ethnicity. One of `white`, `Black`, `Native`, or `NaN` (= unknown). Always assigned via biographical research. Not very useful, as the values are almost exclusively

white or NaN. This fact tells you something about the US literary field in the nineteenth century.

- `occupation` and `occupation_free`: The author's primary employment identification. Recall that the US in the nineteenth century didn't always have a large market for novels, so many of the authors in the corpus made their living by other means. The difference between these fields is that `occupation` uses a fixed vocabulary, while `occupation_free` does not (so includes more detailed or fine-grained information).
- `state_*`: The state in which the author was born, died, and with which they are conventionally associated (`main`).
- `born` and `died`: Year of the author's birth and death, respectively, where known.

✓ 2. Data and methods

```
!git clone https://github.com/wilkens-teaching/info3350-f24
```

 fatal: destination path 'info3350-f24' already exists and is not an empty directory.

```
# Imports
import os
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from collections import Counter
from scipy.stats import entropy
from gensim import corpora, models
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud

nltk.download('punkt_tab')
nltk.download('stopwords')
nltk.download('wordnet')

stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

# File locations
# Note that metadata are supplied as a TSV file
# Text files are in a directory, one file per (long, novel-like) document
```

```

[ntlk_data] Downloading package punkt_tab to /root/nltk_data...
[ntlk_data] Package punkt_tab is already up-to-date!
[ntlk_data] Downloading package stopwords to /root/nltk_data...
[ntlk_data] Package stopwords is already up-to-date!
[ntlk_data] Downloading package wordnet to /root/nltk_data...
[ntlk_data] Package wordnet is already up-to-date!

```

Start coding or [generate](#) with AI.

```

# Load the metadata
metadata = pd.read_csv('/content/info3350-f24/data/us_fiction/corpus_data.tsv', sep='\t', low
file_column = [col for col in metadata.columns if 'source' in col.lower() and 'id' in col.lc

import os

text_dir = '/content/info3350-f24/data/us_fiction/us_texts/'

def load_texts(text_dir):
    texts = {}
    for file_name in os.listdir(text_dir):
        if not file_name.startswith('.'):
            with open(os.path.join(text_dir, file_name), 'r', encoding='utf-8', errors='ignc
                texts[file_name] = file.read()
    return texts

text_files = load_texts(text_dir)
print(f"Loaded {len(text_files)} text files.")

for file_name, content in list(text_files.items())[:1]:
    print(f"{file_name}: {content[:500]}...")

```

```

Loaded 1540 text files.
Wright2-0166: INTRODUCTORY.
I AM not young, may be I am not old enough, to be indulged in the prosiness that I propo

```



✓ Preprocessing the data and extract food related content

```

food_terms = [
    # General foods
    'bread', 'meat', 'corn', 'rice', 'soup', 'fruit', 'vegetable', 'fish', 'egg', 'potato',

    # Beverages
    'tea', 'coffee', 'wine', 'ale', 'cider', 'milk', 'beer', 'whiskey', 'brandy', 'gin', 'ru

    # Desserts and sweets
    'cake', 'pie', 'pudding', 'tart', 'pastry', 'sugar', 'cream', 'candy', 'honey', 'chocola

```

```

# Meals
'breakfast', 'dinner', 'supper', 'feast', 'banquet', 'lunch', 'repast', 'snack', 'tea-ti

# Specific meats and preparations
'roast', 'bacon', 'ham', 'steak', 'pork', 'chicken', 'beef', 'turkey', 'sausage', 'muttc

# Other food-related terms
'butter', 'cheese', 'loaf', 'flour', 'stew', 'gravy', 'broth', 'biscuit', 'porridge', 'p

# Drinks and side items
'juice', 'lemonade', 'syrup', 'water', 'brew', 'yeast', 'custard', 'waffle', 'dumpling',
]
import re

```

```

def extract_food_sentences_fast(text, food_terms):
    sentences = re.split(r'[.!?]', text.lower())
    food_sentences = [sentence for sentence in sentences if any(food in sentence for food in food_terms)]
    return ' '.join(food_sentences)

```

```

processed_texts = {file_name: extract_food_sentences_fast(content, food_terms)
                    for file_name, content in text_files.items()}

```

```

def clean_file_name(file_name):
    file_name = str(file_name)
    return file_name.replace('.txt', '').lower()

```

```

metadata['food_text'] = metadata.index.map(lambda x: processed_texts.get(clean_file_name(x), ''))

```

Grouping and structuring the data

```

metadata = metadata.reset_index() # Reset index to expose source_id column
metadata['source_id'] = metadata['source_id'].astype(str).str.lower() # Ensure string format

# Map food-related text to metadata using source_id
metadata['food_text'] = metadata['source_id'].map(lambda x: processed_texts.get(f"{x}", ""))
#new timeperiod column
metadata['time_period'] = metadata['pub_date'].apply(lambda x: 'Pre-Civil War' if x < 1861 else 'Post-Civil War')

# Verify the new column
print(metadata[['pub_date', 'time_period']].head())
# Verify the updated metadata
print(metadata[['source_id', 'food_text']].head())

```

```

➡ pub_date    time_period
0      1841  Pre-Civil War

```


	source_id	food_text
1	1831	Pre-Civil War
2	1838	Pre-Civil War
3	1839	Pre-Civil War
4	1792	Pre-Civil War
0	eaf001	his talents were of a\nhigh order he had loo...
1	eaf002	\nit\nwas a bitter afternoon in december, the\...
2	eaf003	\nprize tale\nby delia s bacon \nit\nwas almo...
3	eaf004	\nthese were the evenings last year, when\nthe...
4	eaf005	\noriginal state of the\nforest\n \nthere\nwas...

```
# Check rows with non-empty 'food_text'
non_empty_food_text = metadata[metadata['food_text'].str.strip() != '']
print(f"Number of rows with non-empty food text: {len(non_empty_food_text)}")
print(non_empty_food_text[['source_id', 'food_text']].head())
```

➞ Number of rows with non-empty food text: 488

	source_id	food_text
0	eaf001	his talents were of a\nhigh order he had loo...
1	eaf002	\nit\nwas a bitter afternoon in december, the\...
2	eaf003	\nprize tale\nby delia s bacon \nit\nwas almo...
3	eaf004	\nthese were the evenings last year, when\nthe...
4	eaf005	\noriginal state of the\nforest\n \nthere\nwas...

compute token frequencies

```
from collections import Counter
import nltk
from nltk.tokenize import word_tokenize

# Function to calculate food term frequencies
def calculate_token_frequencies(text, food_terms):
    tokens = word_tokenize(text.lower()) # Tokenize text into words
    return Counter([token for token in tokens if token in food_terms])

# Apply token frequency analysis to food_text
metadata['food_word_counts'] = metadata['food_text'].apply(
    lambda x: calculate_token_frequencies(x, food_terms) if pd.notnull(x) else Counter()
)

# Filter out rows with empty counters
metadata = metadata[metadata['food_word_counts'].apply(len) > 0]

# Function to merge Counter objects
def merge_counters(counter_list):
    total_counter = Counter()
    for counter in counter_list:
        total_counter += counter # Merge counters
    return total_counter
```

```
# Group data by region, gender, and time period
grouped_data = (
    metadata.groupby(['pub_place', 'gender', 'time_period'])['food_word_counts']
    .apply(merge_counters)
    .reset_index()
)
```

```
# Display grouped data with aggregated token frequencies
print("Grouped Food Term Frequencies:")
print(grouped_data.head())
```

```
⇒ Grouped Food Term Frequencies:
```

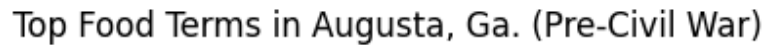
	pub_place	gender	time_period	level_3	food_word_counts
0	Augusta, [Ga.]	M	Pre-Civil War	breakfast	6.0
1	Augusta, [Ga.]	M	Pre-Civil War	dinner	7.0
2	Augusta, [Ga.]	M	Pre-Civil War	repast	3.0
3	Augusta, [Ga.]	M	Pre-Civil War	meat	6.0
4	Augusta, [Ga.]	M	Pre-Civil War	corn	5.0

bar chart visualization

```
import matplotlib.pyplot as plt

# Example: Plot top food terms for Augusta, Ga.
region_data = grouped_data[grouped_data['pub_place'] == 'Augusta, [Ga.]']
region_data = region_data.set_index('level_3')['food_word_counts']

plt.figure(figsize=(8, 6))
region_data.plot(kind='bar')
plt.title("Top Food Terms in Augusta, Ga. (Pre-Civil War)")
plt.ylabel("Frequency")
plt.xlabel("Food Terms")
plt.show()
```



```
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.title("Word Cloud of Food Terms in Augusta, Ga.")
plt.show()
```



Word Cloud of Food Terms in Augusta, Ga.



KL divergence

```
from collections import Counter

# Function to merge Counters
def merge_counters(counter_list):
    total_counter = Counter()
    for counter in counter_list:
        total_counter += counter # Merge counters
    return total_counter

# Select two regions for comparison
region1 = 'Augusta, [Ga.]'
region2 = 'Albany [N.Y.]'

# Extract and merge food_word_counts for each region
food_counts1 = metadata[metadata['pub_place'] == region1]['food_word_counts'].apply(Counter)
food_counts1 = merge_counters(food_counts1)

food_counts2 = metadata[metadata['pub_place'] == region2]['food_word_counts'].apply(Counter)
food_counts2 = merge_counters(food_counts2)

# Normalize Counter objects (convert counts to probabilities)
def normalize_counter(counter):
    total = sum(counter.values())
    return {key: value / total for key, value in counter.items()}
```

```
# Function to compute KL Divergence
from scipy.stats import entropy
import numpy as np

def kl_divergence(counter1, counter2):
    dist1 = normalize_counter(counter1)
    dist2 = normalize_counter(counter2)
    vocab = set(dist1.keys()).union(set(dist2.keys()))
    p = np.array([dist1.get(term, 1e-10) for term in vocab])
    q = np.array([dist2.get(term, 1e-10) for term in vocab])
    return entropy(p, q)

# Compute KL Divergence
kl_score = kl_divergence(food_counts1, food_counts2)
print(f"KL Divergence between {region1} and {region2}: {kl_score:.4f}")
```

→ KL Divergence between Augusta, [Ga.] and Albany [N.Y.]: 0.3578

Systematic comparison of groups

```
import itertools

# Create a list of unique regions
regions = metadata['pub_place'].unique()

# Compare all pairs of regions
kl_results = []

for region1, region2 in itertools.combinations(regions, 2):
    food_counts1 = metadata[metadata['pub_place'] == region1]['food_word_counts'].apply(Cour
    food_counts1 = merge_counters(food_counts1)

    food_counts2 = metadata[metadata['pub_place'] == region2]['food_word_counts'].apply(Cour
    food_counts2 = merge_counters(food_counts2)

    # Compute KL Divergence
    kl_score = kl_divergence(food_counts1, food_counts2)
    kl_results.append((region1, region2, kl_score))

# Convert results to a DataFrame for easier analysis
kl_df = pd.DataFrame(kl_results, columns=['Region 1', 'Region 2', 'KL Divergence'])

# Display the top results
print("KL Divergence Between Food Term Distributions (Regions):")
print(kl_df.sort_values(by='KL Divergence', ascending=False).head())
```

→ KL Divergence Between Food Term Distributions (Regions):

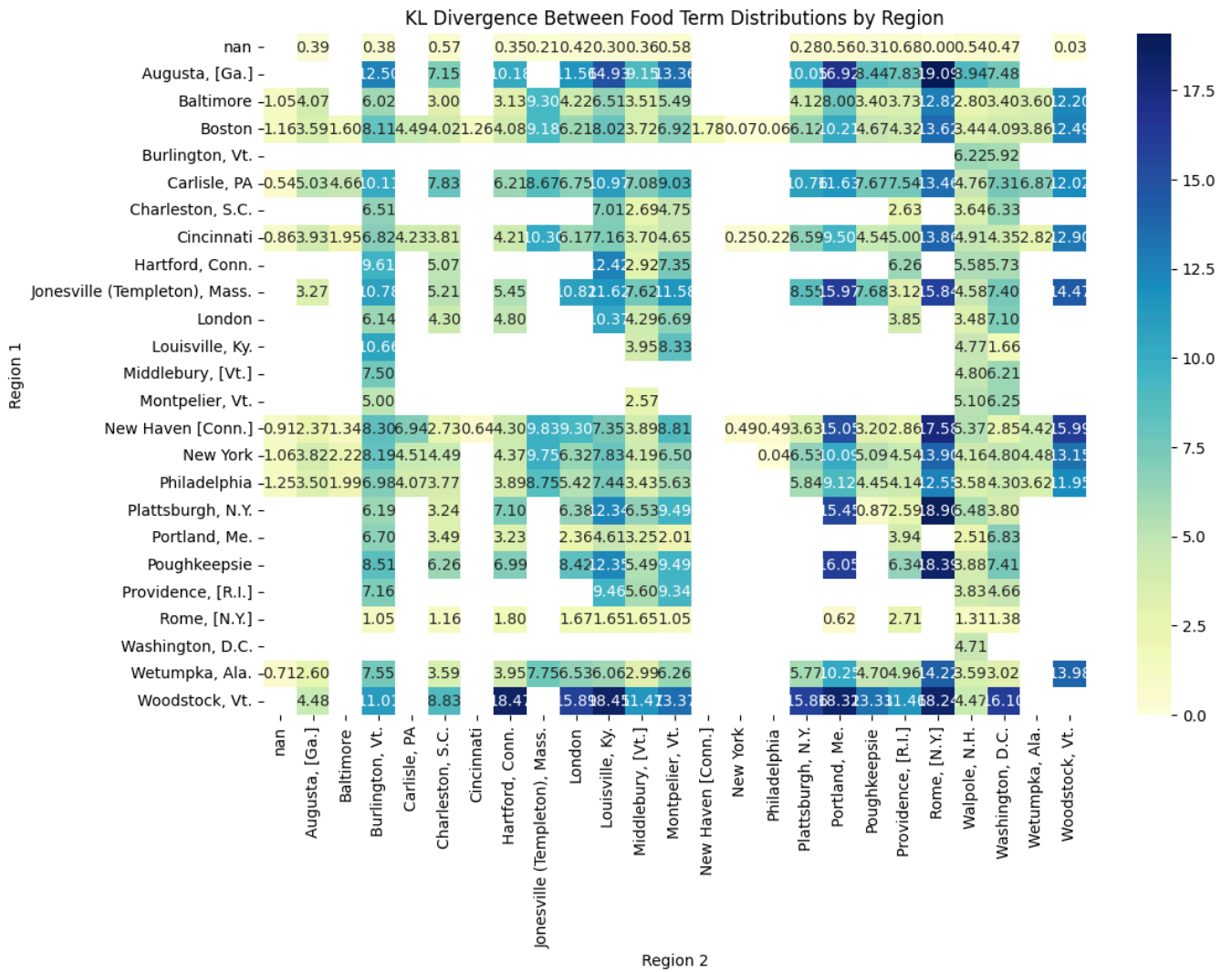
	Region 1	Region 2	KL Divergence
222	Augusta, [Ga.]	Rome, [N.Y.]	19.087821

235	Plattsburgh, N.Y.	Rome, [N.Y.]	18.896494
211	Woodstock, Vt.	Hartford, Conn.	18.466816
214	Woodstock, Vt.	Louisville, Ky.	18.447547
247	Poughkeepsie	Rome, [N.Y.]	18.386803

KL divergence as heatmap

```
# Create a pivot table for the heatmap
kl_pivot = kl_df.pivot(index="Region 1", columns="Region 2", values="KL Divergence")

# Generate a heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(kl_pivot, annot=True, cmap="YlGnBu", fmt=".2f")
plt.title("KL Divergence Between Food Term Distributions by Region")
plt.show()
```



comparing genders and time periods

```
genders = metadata['gender'].unique()
```

```
# Compare all pairs of genders
gender_results = []
```

```
for gender1, gender2 in itertools.combinations(genders, 2):
    food_counts1 = metadata[metadata['gender'] == gender1]['food_word_counts'].apply(Counter)
    food_counts1 = merge_counters(food_counts1)
```

```

food_counts2 = metadata[metadata['gender'] == gender2]['food_word_counts'].apply(Counter)
food_counts2 = merge_counters(food_counts2)

kl_score = kl_divergence(food_counts1, food_counts2)
gender_results.append((gender1, gender2, kl_score))

```

Display results

```

gender_df = pd.DataFrame(gender_results, columns=['Gender 1', 'Gender 2', 'KL Divergence'])
print("KL Divergence Between Food Term Distributions (Genders):")
print(gender_df)

```

```

↗ KL Divergence Between Food Term Distributions (Genders):
  Gender 1 Gender 2 KL Divergence
0        M        F         0.137979
1        M      NaN         1.157715
2        F      NaN         0.914057

```

Start coding or [generate](#) with AI.

```

import itertools
from collections import Counter

```

```
region_results = []
```

```

for region1, region2 in itertools.combinations(metadata['pub_place'].unique(), 2):
    food_counts1 = metadata[metadata['pub_place'] == region1]['food_word_counts'].apply(Counter)
    food_counts1 = merge_counters(food_counts1)

    food_counts2 = metadata[metadata['pub_place'] == region2]['food_word_counts'].apply(Counter)
    food_counts2 = merge_counters(food_counts2)

    kl_score = kl_divergence(food_counts1, food_counts2)
    region_results.append((region1, region2, kl_score))

```

Convert to DataFrame and display

```

region_df = pd.DataFrame(region_results, columns=['Region 1', 'Region 2', 'KL Divergence'])
print(region_df.sort_values(by='KL Divergence', ascending=False))

```

```

↗
   Region 1      Region 2  KL Divergence
222  Augusta, [Ga.]  Rome, [N.Y.]      19.087821
235  Plattsburgh, N.Y.  Rome, [N.Y.]      18.896494
211  Woodstock, Vt.  Hartford, Conn.      18.466816
214  Woodstock, Vt.  Louisville, Ky.      18.447547
247  Poughkeepsie    Rome, [N.Y.]      18.386803
..      ...      ...
2      Boston      New York      0.068474
3      Boston      Philadelphia      0.061660
72     New York      Philadelphia      0.041522
173      NaN      Woodstock, Vt.      0.034510
177      NaN      Rome, [N.Y.]      0.000000

```


[325 rows x 3 columns]

Fighting words Analysis for comparison of gender

```
import pandas as pd
import numpy as np
from collections import Counter
from scipy.stats import norm
import matplotlib.pyplot as plt
import seaborn as sns
from nltk.tokenize import word_tokenize

# Function to compute log-odds ratio with prior
def compute_log_odds(group1_counts, group2_counts, prior_counts, alpha=1):
    """
    Compute log-odds ratio between two groups with an informative prior.
    """
    # Convert to counters if input is not already a Counter
    group1_counts = Counter(group1_counts)
    group2_counts = Counter(group2_counts)

    vocab = set(group1_counts.keys()).union(set(group2_counts.keys()))
    log_odds = {}

    # Compute prior counts
    prior_sum = sum(prior_counts.values()) + alpha * len(vocab)

    for word in vocab:
        # Calculate counts with priors
        count1 = group1_counts[word] + alpha * prior_counts[word] + 1
        count2 = group2_counts[word] + alpha * prior_counts[word] + 1

        # Total counts for each group
        total1 = sum(group1_counts.values()) + prior_sum
        total2 = sum(group2_counts.values()) + prior_sum

        # Log-odds ratio
        log_odds[word] = np.log(count1 / total1) - np.log(count2 / total2)

    return log_odds

# Tokenize and preprocess food text for fighting words
def tokenize_food_text(metadata, group_column):
    """
    Tokenize 'food_text' column and group tokens by the specified group_column.
    """
    grouped_tokens = {}
    for group, data in metadata.groupby(group_column):
        tokens = []
```

```

    for text in data['food_text']:
        tokens.extend(word_tokenize(text.lower()))
    grouped_tokens[group] = Counter(tokens)
    return grouped_tokens

# Merge counters to get the prior
def compute_prior_counts(grouped_tokens):
    prior_counts = Counter()
    for group, tokens in grouped_tokens.items():
        prior_counts += tokens
    return prior_counts

# Fighting Words Analysis
def fighting_words_analysis(metadata, group_column):
    """
    Perform Fighting Words analysis between two groups.
    """
    # Tokenize food_text and group tokens by the specified column
    grouped_tokens = tokenize_food_text(metadata, group_column)
    prior_counts = compute_prior_counts(grouped_tokens)

    # Select two groups for comparison
    groups = list(grouped_tokens.keys())
    if len(groups) != 2:
        print(f"Error: Fighting Words requires exactly 2 groups. Found {len(groups)} groups.")
        return

    group1, group2 = groups[0], groups[1]
    print(f"Comparing Fighting Words between: {group1} and {group2}")

    # Compute log-odds ratio
    log_odds = compute_log_odds(grouped_tokens[group1], grouped_tokens[group2], prior_counts)

    # Sort log-odds to find the most distinctive words
    sorted_log_odds = sorted(log_odds.items(), key=lambda x: x[1], reverse=True)
    return group1, group2, sorted_log_odds

def fighting_words_food_terms(metadata, group_column, food_terms, top_n=10):
    """
    Perform Fighting Words Analysis restricted to food terms only.
    """
    # Perform Fighting Words Analysis
    group1, group2, log_odds = fighting_words_analysis(metadata, group_column)

    # Ensure log_odds is a dictionary (convert if needed)
    if isinstance(log_odds, list):
        log_odds = dict(log_odds)

```

```
# Filter log odds ratios to include only food terms
food_log_odds = {word: score for word, score in log_odds.items() if word in food_terms}

# Sort by log odds values and select top N
sorted_food_terms = sorted(food_log_odds.items(), key=lambda x: abs(x[1]), reverse=True)

# Prepare data for visualization
words, scores = zip(*sorted_food_terms)
colors = ['blue' if score > 0 else 'red' for score in scores] # Color coding for groups

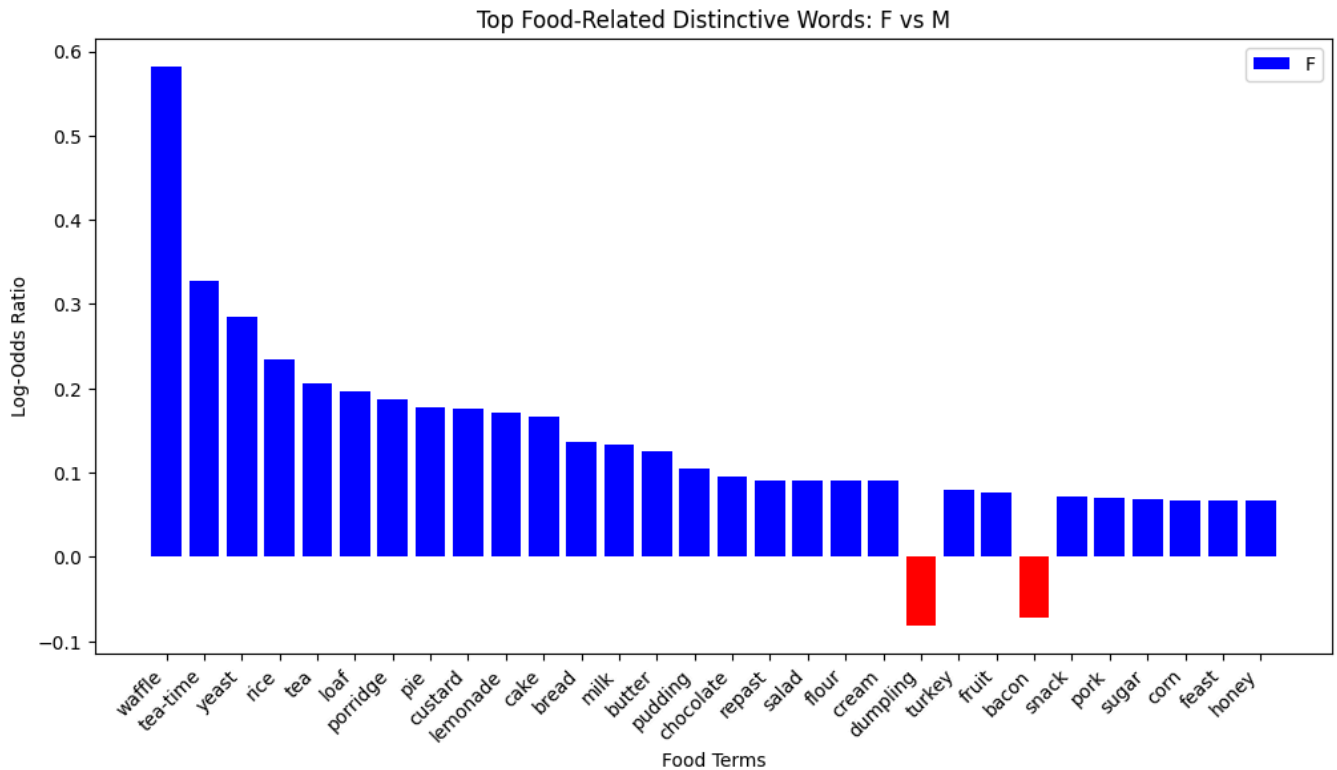
# Plot
plt.figure(figsize=(12, 6))
plt.bar(words, scores, color=colors)
plt.xticks(rotation=45, ha="right")
plt.xlabel("Food Terms")
plt.ylabel("Log-Odds Ratio")
plt.title(f"Top Food-Related Distinctive Words: {group1} vs {group2}")
plt.legend(labels=[group1, group2], loc="upper right")
plt.show()

return group1, group2, food_log_odds

# Usage
group1, group2, food_log_odds = fighting_words_food_terms(
    metadata=metadata,
    group_column='gender',
    food_terms=food_terms,
    top_n=30
)
```



Comparing Fighting Words between: F and M



fighting words analysis for region

```
def compute_log_odds(counter1, counter2, alpha=0.01):
    """
    Calculate log-odds ratios for words between two Counter objects.

    :param counter1: Word counts for group 1.
    :param counter2: Word counts for group 2.
    :param alpha: Smoothing parameter to avoid division by zero.
    :return: DataFrame with words, log-odds, and group labels.
    """
    vocab = set(counter1.keys()).union(set(counter2.keys()))
    words, log_odds, groups = [], [], []

    for word in vocab:
        # Smoothed counts
        count1 = counter1.get(word, 0) + alpha
        count2 = counter2.get(word, 0) + alpha

        # Log-odds calculation
```

```

log_odds_ratio = np.log(count1 / count2)
words.append(word)
log_odds.append(log_odds_ratio)
groups.append("Group 1" if log_odds_ratio > 0 else "Group 2")

return pd.DataFrame({"word": words, "log_odds": log_odds, "group": groups})

northern_states = [
    "Connecticut", "Delaware", "Illinois", "Indiana", "Iowa", "Kansas",
    "Maine", "Massachusetts", "Michigan", "Minnesota", "New Hampshire",
    "New Jersey", "New York", "Ohio", "Oregon", "Pennsylvania", "Rhode Island",
    "Vermont", "Wisconsin"
]

southern_states = [
    "Alabama", "Arkansas", "Florida", "Georgia", "Kentucky", "Louisiana",
    "Maryland", "Mississippi", "Missouri", "North Carolina", "South Carolina",
    "Tennessee", "Texas", "Virginia", "West Virginia"
]

words_region1 = merge_counters(metadata[metadata['pub_place'] == region1]['food_word_counts']
words_region2 = merge_counters(metadata[metadata['pub_place'] == region2]['food_word_counts']
region_words = compute_log_odds(words_region1, words_region2)
# Example usage with Fighting Words DataFrame
# Assume 'region_words' DataFrame has 'word', 'log_odds', and 'group' columns
region_words = compute_log_odds(words_region1, words_region2)

def plot_fighting_words(fighting_words, group1, group2, title="Region Comparison"):
    """
    Generate a bar chart for Fighting Words results.

    :param fighting_words: DataFrame with 'word', 'log_odds', and 'group'.
    :param group1: Name of the first group (e.g., "North").
    :param group2: Name of the second group (e.g., "South").
    :param title: Title for the plot.
    """
    # Sort by absolute log-odds to display top words
    fighting_words = fighting_words.sort_values(by='log_odds', ascending=False)

    # Plot top 10 positive and negative log-odds words
    plt.figure(figsize=(12, 6))
    sns.barplot(x='word', y='log_odds', hue='group', data=fighting_words.head(20))

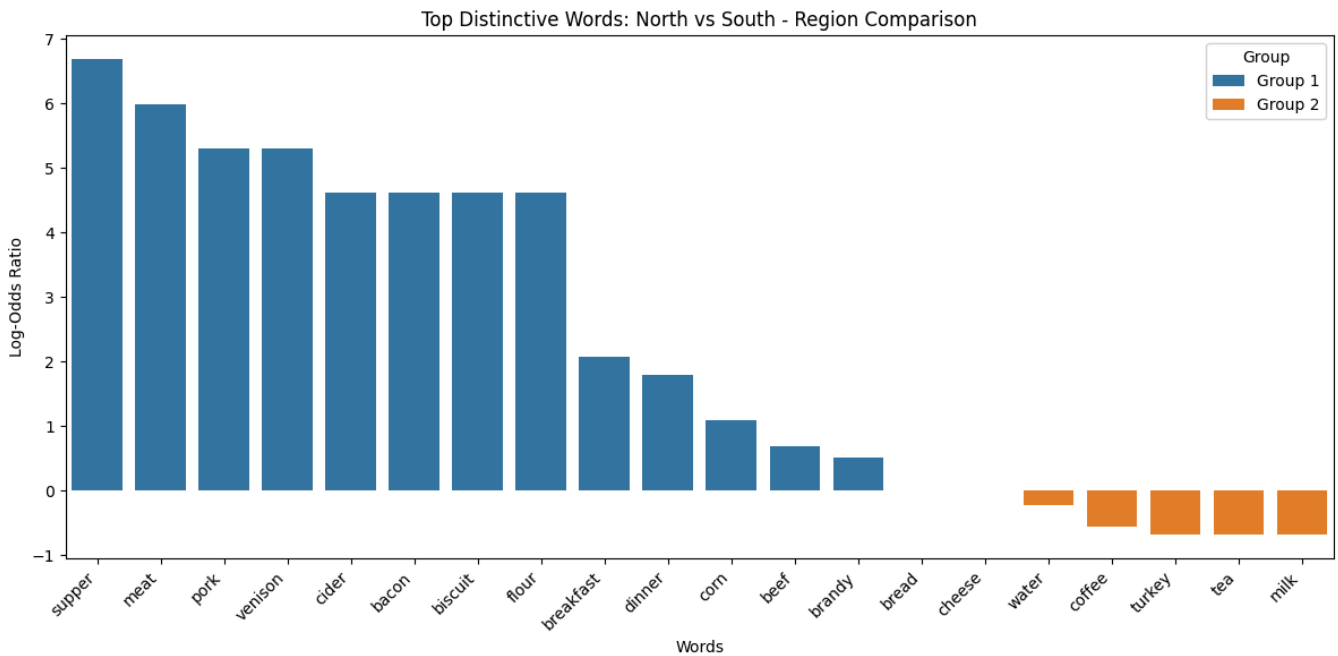
    # Add titles and labels
    plt.title(f"Top Distinctive Words: {group1} vs {group2} - {title}")
    plt.ylabel("Log-Odds Ratio")
    plt.xlabel("Words")
    plt.xticks(rotation=45, ha="right")
    plt.legend(title="Group")
    plt.tight_layout()

```

```
plt.show()
```

```
# Example usage for regions
```

```
plot_fighting_words(region_words, group1="North", group2="South", title="Region Comparison")
```

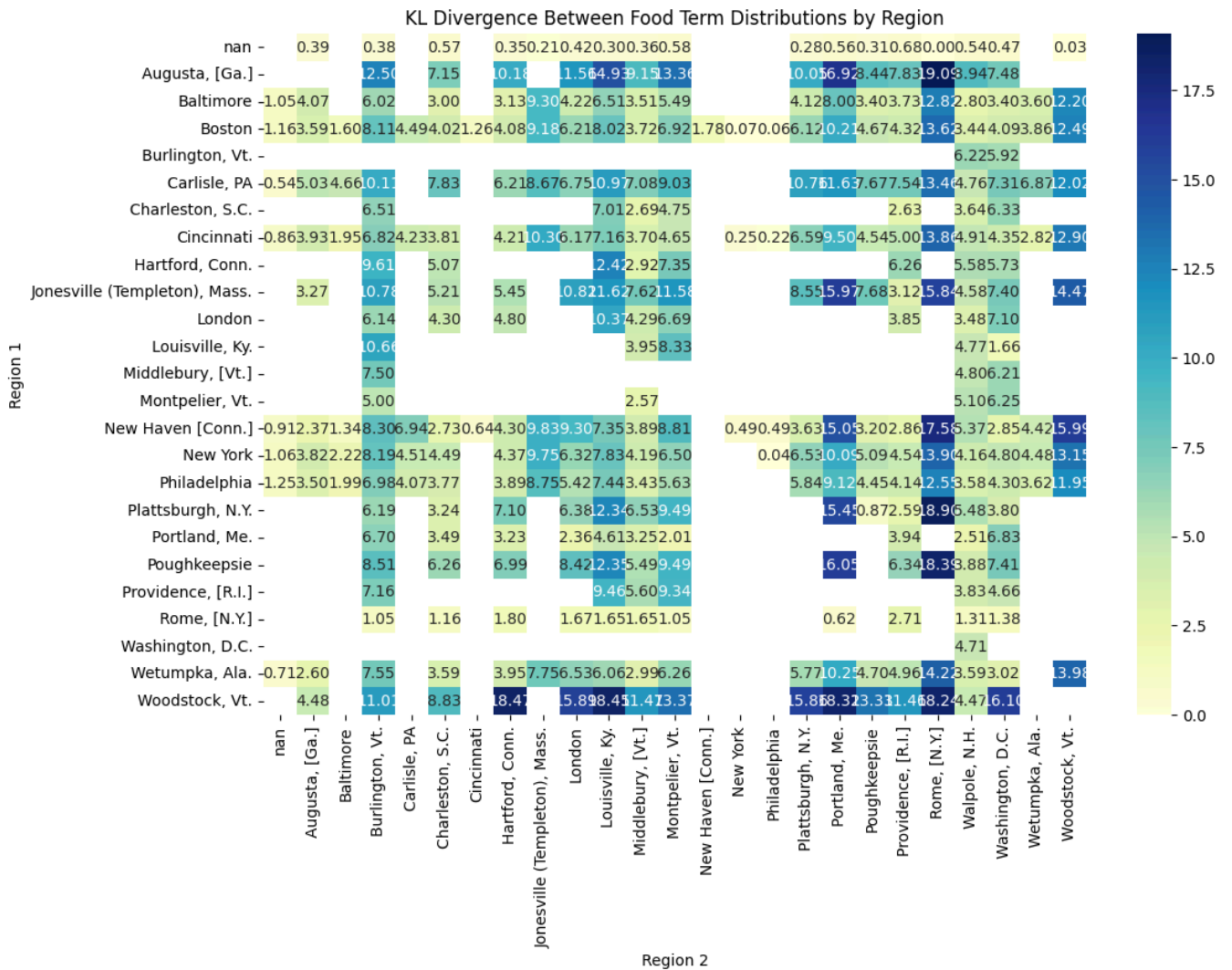


✓ 3. Results

In this section, we present the results of the analyses conducted on the 19th-century literary corpus. Key areas of exploration include **regional comparisons**, **gender comparisons**, and **temporal (time period) comparisons** of food-related terms. We also include a **Fighting Words Analysis** to identify distinctive words between different groups.

✓ 1. Regional Comparison of Food Term Distributions

```
plt.figure(figsize=(12, 8))
sns.heatmap(kl_pivot, annot=True, cmap="YlGnBu", fmt=".2f")
plt.title("KL Divergence Between Food Term Distributions by Region")
plt.show()
```



To assess how food-related vocabularies varied across regions, we computed the **KL Divergence** for all pairs of regions. Higher KL divergence indicates greater dissimilarity in food term distributions.

The heatmap below visualizes these results:

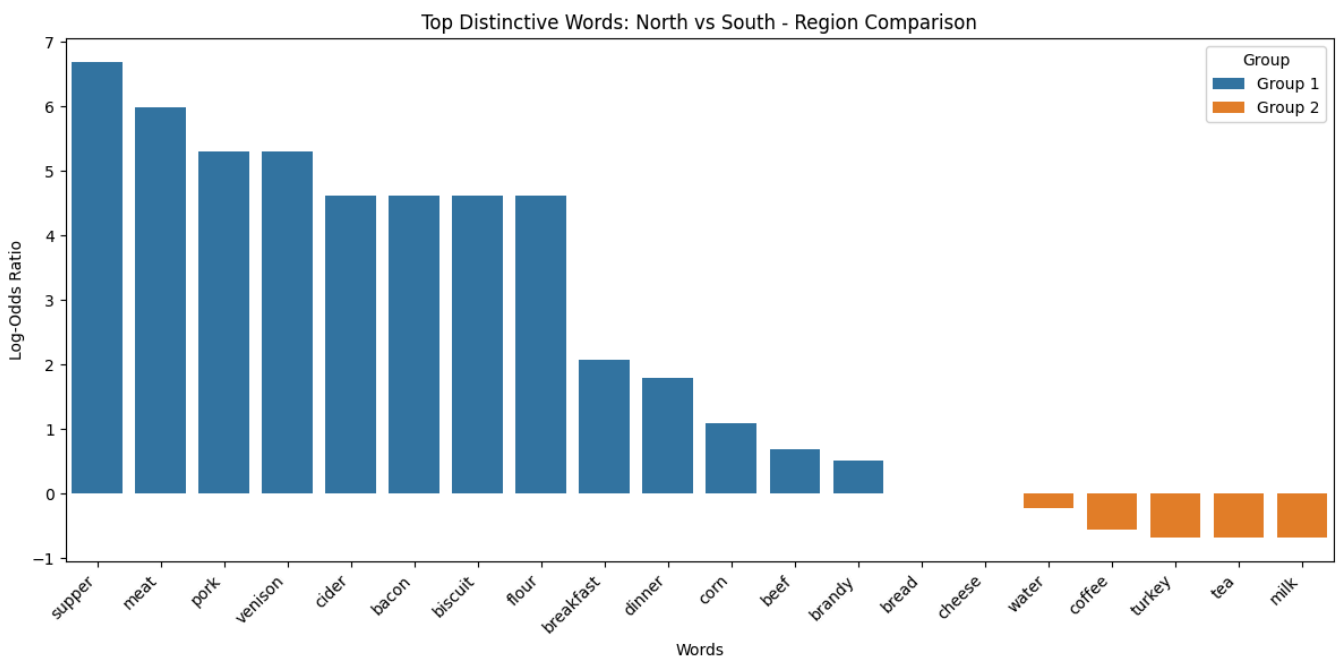
- Darker blue colors represent regions with **high divergence**, indicating unique food vocabularies.
- Lighter colors suggest more similar food vocabularies.

Key Observations:

- Significant divergence is observed between **Augusta, Ga.** and **Rome, N.Y.** (KL Divergence ~19).
- Northern regions like **New York** and **Philadelphia** exhibit more similarities.
- Southern regions, such as Augusta, tend to diverge strongly from Northern cities.

Top Distinctive Words North Vs South Region Comparison fighting words

```
plot_fighting_words(region_words, group1="North", group2="South", title="Region Comparison")
```



We applied Fighting Words Analysis to identify food-related terms that are significantly more distinctive in the North versus the South. Using the log-odds ratio, the most prominent food-related words for each region were highlighted.

Findings

Northern Region: The Northern region demonstrated a strong association with several food terms, with "supper" showing the highest log-odds ratio (~6.8), followed by "meat", "pork", and "venison".

These words suggest a regional preference for hearty meals and specific types of meat. Additionally, terms like "flour", "biscuit", and "cider" further indicate a connection to baked goods and traditional food preparation practices.

Southern Region: In contrast, the Southern region exhibited a distinctive usage of terms such as "milk", "turkey", "tea", and "coffee". The prominence of "tea" and "coffee" suggests the cultural significance of beverages in the South, particularly sweetened tea as a regional staple.

Visual Insights

The bar chart above visually captures the distinctiveness of food terms by region:

Words associated with the Northern region are represented in blue and dominate the upper half of the chart with higher log-odds ratios. Words linked to the Southern region, represented in orange, show lower log-odds ratios but remain significant for distinguishing regional food patterns.

Interpretation

The results highlight a divergence in food culture between the North and South. The Northern regions favor terms indicative of meals and specific meats ("supper," "meat," "bacon," "venison"), aligning with historical patterns of rural and agricultural food practices. On the other hand, the Southern regions emphasize beverages ("tea," "coffee") and broader food staples ("milk," "turkey"), reflecting a cultural inclination toward hospitality and shared dining practices.

✓ Gender comparison of food terms

Here I use KL Divergence to compare the food-related vocabularies between **male** and **female** authors used.

Distinctive Words by Gender

```
from collections import Counter
import numpy as np

def compute_log_odds(group1_counts, group2_counts, prior_counts, alpha=1):
    """
    Compute log-odds ratio between two groups with an informative prior.
    """
    # Convert to counters if input is not already a Counter
    group1_counts = Counter(group1_counts)
    group2_counts = Counter(group2_counts)

    vocab = set(group1_counts.keys()).union(set(group2_counts.keys()))
    log_odds = {}

    # Compute prior counts
```

```
prior_sum = sum(prior_counts.values()) + alpha * len(vocab)

for word in vocab:
    # Calculate counts with priors
    count1 = group1_counts[word] + alpha * prior_counts[word] + 1
    count2 = group2_counts[word] + alpha * prior_counts[word] + 1

    # Total counts for each group
    total1 = sum(group1_counts.values()) + prior_sum
    total2 = sum(group2_counts.values()) + prior_sum

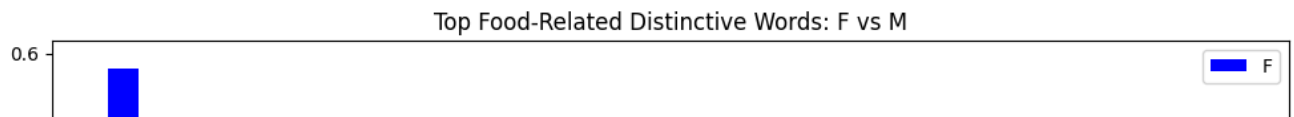
    # Log-odds ratio
    log_odds[word] = np.log(count1 / total1) - np.log(count2 / total2)

return log_odds

group1, group2, food_log_odds = fighting_words_food_terms(
    metadata=metadata,
    group_column='gender',
    food_terms=food_terms,
    top_n=30
)
```



Comparing Fighting Words between: F and M



The Fighting Words Analysis was applied to examine differences in food-related language between texts authored or associated with F (Female) and M (Male) groups. The analysis identified food terms with significantly distinctive log-odds ratios for each group.

Findings Distinctive Food Terms for Female-Associated Texts: The majority of food terms are distinctive to the Female group, as seen in the positive log-odds ratios (blue bars). Notable terms include:

"waffle" (highest log-odds ratio ~0.6) "tea-time", "yeast", "rice", and "tea" These terms suggest an association with baking, beverages, and meals traditionally linked to domestic settings and refined dining.

Distinctive Food Terms for Male-Associated Texts: A smaller number of food terms are distinctive to the Male group, as indicated by the negative log-odds ratios (red bars). Prominent terms include:

"turkey", "fruit", and "bacon" These terms highlight a focus on meat and general food staples, aligning with a more protein-heavy or pragmatic diet portrayal.

Visual Insights The bar chart illustrates the log-odds ratios of food terms: