# Comprehensive NLP Analysis of Airbnb Reviews for Improved Decision-Making

George Washington University

**Final Term Project - Individual Final Report**

**Author**

**Kusum Sai Chowdary Sannapaneni**
Department of Data Science
George Washington University
kusumsaichowdary.sannapaneni@gwu.edu

## Abstract

This project explores methods to enhance sentiment and aspect-based analysis of Airbnb guest reviews using advanced Natural Language Processing (NLP) techniques. By employing transformers, aspect extraction, and summarization, we aim to improve the efficiency of feedback interpretation. Our approach utilizes transformer-based models, including BERT and T5, for sentiment classification and summarization, alongside Word2Vec and KMeans clustering for aspect-based sentiment analysis (ABSA). The study evaluates the influence of aspect granularity, dataset diversity, and pre-trained models on overall performance. Results are expected to demonstrate that combining transformer-based models with aspect-specific features enhances predictive accuracy and provides actionable insights for hosts. This project contributes to the field of text analysis by offering a framework to automate and refine the interpretation of large-scale user-generated content for short-term rental platforms.

## 2 Introduction

The rapid growth of platforms like Airbnb has generated vast user reviews offering insights into guest satisfaction, property conditions, and host performance. However, analyzing these reviews manually is inefficient. This report leverages Natural Language Processing (NLP) techniques to automate insights extraction through (1) sentiment analysis, (2) aspect-based sentiment analysis (ABSA), and (3) summarization. These methods aim to help hosts, guests, and platform administrators improve guest experiences and address recurring issues effectively.

## 3 Project Goals

This project comprehensively analyzes NYC Airbnb reviews (2021) using advanced NLP techniques. Specifically, we:

- **Preprocess and Sentiment Analysis (Phase 1):** Prepare and fine-tune a BERT model to classify reviews as positive or negative.
- **Aspect-Based Sentiment Analysis (Phase 2):** Identify and cluster key aspects (e.g., location, amenities) and map sentiments to them.
- **Summarization (Phase 3):** Generate concise summaries of multiple reviews using a transformer-based model (T5), thereby aiding quick decision-making.

## 4 Dataset Description

The dataset consists of 17,444 Airbnb reviews from New York City (2021). Key columns include:

- listing_id: Unique identifier for each listing.
- url: Airbnb listing page link.
- review_posted_date: Review date in "Month Year" format.
- review: The free-text guest review.

All analyses focus on the review text after ensuring it is in English and well-preprocessed.

## 5 My Role and Shared Work

- I primarily focused on **Phase 1: Sentiment Analysis and Data Preprocessing**. This included
    - Comprehensive data cleaning (removing non-English reviews, expanding contractions, handling emojis, normalizing text).
    - Initial sentiment labeling using a pre-trained Hugging Face DistilBERT model.
    - Fine-tuning a BERT model for improved sentiment classification accuracy.
    - Generating initial visualizations for sentiment distribution and time-series trends.
    - **Phase 2: ABSA: Word2Vec and KMeans clustering Approach and coding.**
    - **Phase 3: Suggestions for prompt and Chunk Division Logic Building**
- My teammate contributed to
    - Additional visualization methods (bigrams, word clouds).
    - Experimentation with aspect extraction using spaCy (discarded approach) and eventual adoption of Word2Vec and KMeans clustering.
    - Summarization (Phase 3) using T5, including chunking strategies and prompt design.
- Our combined collaboration includes ABSA steps, aspect mapping, and design decisions for final summarization.

## 6 Description of My Individual Work and Methodology

### 6.1 Data Loading and Preprocessing

#### Step 1: Load Data

- Load the dataset NYC_2021_airbnb_reviews_data1.csv into a pandas DataFrame for manipulation.

#### Step 2: Date Parsing

- Converting the review_posted_date column to datetime format which ensures that all reviews have a valid date, which could be useful for temporal analysis.

#### Step 3: Handle Missing Values

- Validated missing reviews which were null, we have none. Missing reviews would not contribute to the sentiment analysis.

**Step 4: Detect Language**

- Used langdetect to identify the language of each review and filters out non-english reviews to ensure consistency, as the sentiment analysis model is trained on english text.

**Step 5: Text Preprocessing**

- **Expanding Contractions:**
    - Used the contractions library to expand words like "don't" to "do not". This method standardizes the text, which helps in better tokenization and model understanding.
- **Cleaning Text:**
    - Removes URLs, emojis (converted to text), mentions, hashtags, and special characters.
- **Further Preprocessing:**
    - **Code:** Tokenize, lemmatize, remove stop words (with custom exceptions), and remove common words that don't contribute to sentiment, improving model performance.

## 6.2 Initial Sentiment Analysis and Fine-Tuning a BERT Model

**Step 6: Initial Sentiment Analysis**

- **Purpose**: To generate sentiment labels for each review in our dataset using a pre-trained sentiment analysis model.
- **Process**:
    - We have used the Hugging Face pipeline with the model 'distilbert-base-uncased-fine tuned-sst-2-english' to predict sentiments.
    - The model assigns a sentiment label ('POSITIVE' or 'NEGATIVE') to each review.
    - These labels are stored in data['initial_sentiment'] and then converted to numerical values (1 for positive, 0 for negative) in data['sentiment_label'].
- **Outcome**: Each review now has an associated sentiment label, which we can use as ground truth for supervised learning.

**Step 7-10: Training Our Own BERT Model**

- **Process**:
    - **Data Splitting**: We split the dataset into training and validation sets, ensuring that the class distribution is maintained (stratify=data['sentiment_label']).
    - **Dataset Preparation**: We tokenize the texts and create Hugging Face Dataset objects for both training and validation data.
    - **Model Initialization**: We initialize a new BertForSequenceClassification model with num_labels=2 to perform binary classification.
    - **Training**: We train this model on our dataset, using the sentiment labels obtained from the initial sentiment analysis as the target labels.
    - **Evaluation**: After training, we evaluate the model's performance on the validation set.

## 6.3 Aspect-Based Sentiment Analysis (ABSA)

While I have not entirely contributed to ASBA, I have my significance here

- The tokenization and lemmatization step for ABSA.
- Training the Word2Vec model and extracting nouns.
- Applying KMeans to cluster noun embeddings into aspects.

**Step 1: Loading Data**

- Process:
    - Reads the CSV file processed_reviews_SentimentLabels.csv into a Pandas DataFrame.
    - Checks if the necessary columns (listing_id, review_posted_date, cleaned_review, sentiment_label) are present.
    - Displays a sample of the data to verify successful loading.

**Step 2: Tokenization and Lemmatization**

- **Process:**
    - Initialize the WordNet lemmatizer.
    - Defines a function to:
        - Convert text to lowercase.
        - Tokenize the text into individual words.
        - Lemmatize the tokens to their base forms.
        - Remove non-alphabetic tokens (e.g., numbers, punctuation).
    - Applies the function to the cleaned_review column to create a new tokens column.

**Step 3: Training the Word2Vec Model**

- **Process:**
    - Converts the list of token lists (sentences) into the format required for Word2Vec training.
    - Initializes and trains the Word2Vec model with specified parameters:
        - vector_size: Dimensionality of the word vectors (100).
        - window: Maximum distance between the current and predicted word
        - min_count: Ignores words with total frequency lower than this (5).
        - workers: Number of worker threads (1 for determinism).
        - epochs: Number of iterations over the corpus (10).
        - seed: Fixed seed for reproducibility (42).

**Step 4: Extracting Nouns from Vocabulary**

- **Process:**
    - Retrieves the vocabulary from the Word2Vec model.
    - Performs Part-of-Speech (POS) tagging on the vocabulary words.
    - Filters out words that are nouns (NN, NNS, NNP, NNPS).
    - Prints the number of nouns extracted and displays a sample.

**Step 5: Obtaining Embeddings for Nouns**

- **Process:**
    - Iterates over the list of nouns.
    - Checks if each noun is present in the Word2Vec model's vocabulary.
    - Retrieves the embedding for the noun and adds it to noun_embeddings.
    - Updates the list of nouns to include only those with embeddings.
    - Converts the list of embeddings into a NumPy array.

**Step 6: Clustering Noun Embeddings**

- **Process:**
    - Sets the number of clusters (num_clusters) to 15.
    - Initializes the KMeans clustering algorithm with the specified number of clusters, fixed seed, and number of initializations.
    - Fits the KMeans model to the noun embeddings and predicts cluster assignments.
    - Creates a DataFrame nouns_clusters with nouns and their assigned clusters.
    - Displays the number of nouns in each cluster.

**Step 7: Assigning Aspect Names to Clusters**

- **Process:**
    - Defines a dictionary manual_cluster_labels mapping cluster numbers to aspect names.
    - Maps the clusters in nouns_clusters to their respective aspect names using the dictionary.
    - Handles any missing aspect assignments by labeling them as 'Other'.
    - Creates a directory clusters_output to save cluster contents for easier inspection.
    - Iterates through each cluster, prints out its nouns, and saves them to separate text files within the clusters_output directory.
    - Displays sample nouns with their assigned aspects.

**Step 8: Associating Aspects with Reviews**

- **Process:**
    - Creates a dictionary noun_to_aspect mapping nouns to their assigned aspects.
    - Defines a function get_aspects_from_tokens to extract aspects from the tokens in a review by checking if any token matches a noun in the noun_to_aspect dictionary.
    - Applies the function to the tokens column to create a new aspects column.

**Step 9: Creating a DataFrame for Aspect-Level Sentiments and saving It as .csv file**

- **Output**: The CSV file AspectbasedSentimentAnalysis.csv contains aspect-level sentiment data.

## 7 Results

### 7.1 Sentiment Distribution Visualization

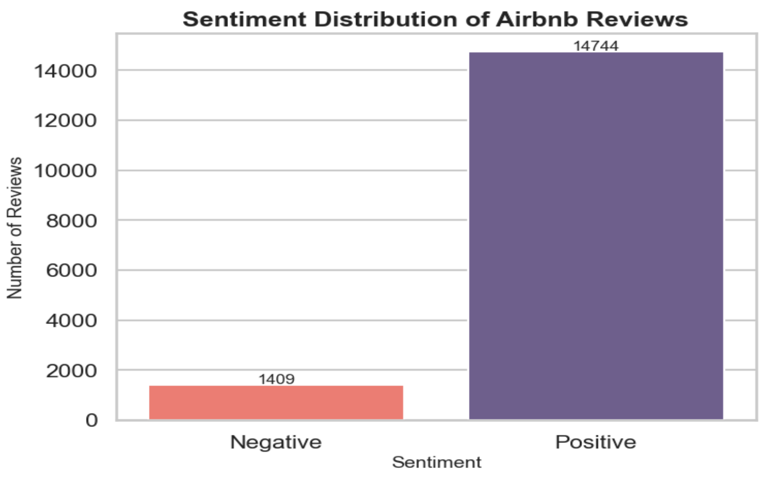


***Fig 1.*** *Bar Chart of Sentiment Distribution*

**Observation:**

- The bar chart showed a significant skew towards positive reviews (**Positive Dominance**), reflecting overall guest satisfaction.
- Negative reviews were comparatively fewer, a common pattern observed in user feedback on hospitality platforms.

### 7.2 Yearly Sentiment Trends

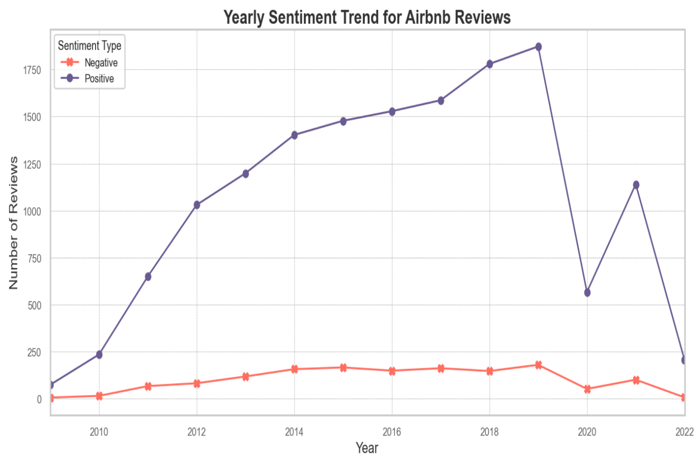
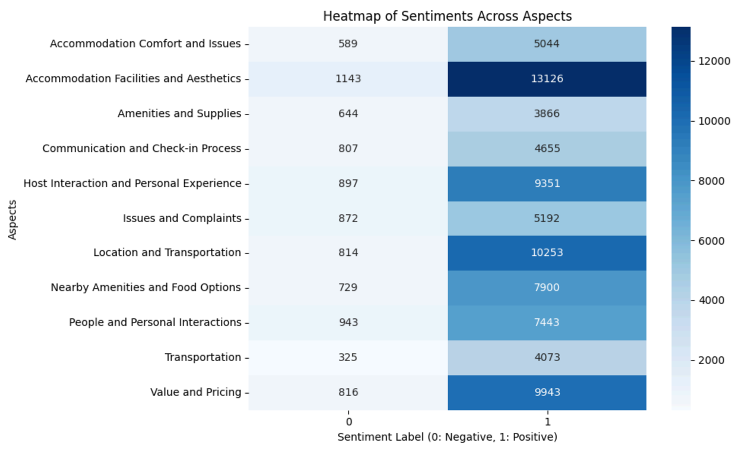


**Fig 2.** ***Line Chart of Yearly Sentiment Counts***

**Observation:**

- Over the analyzed period, positive sentiments remained consistently high.
- Negative sentiment counts were relatively stable and low, indicating consistent service quality or guest satisfaction over time.

## 7.3 Aspect-Based Sentiment (Heatmap)



| Aspects | 0 | 1 |
|---|---|---|
| Accommodation Comfort and Issues | 589 | 5044 |
| Accommodation Facilities and Aesthetics | 1143 | 13126 |
| Amenities and Supplies | 644 | 3866 |
| Communication and Check-in Process | 807 | 4655 |
| Host Interaction and Personal Experience | 897 | 9351 |
| Issues and Complaints | 872 | 5192 |
| Location and Transportation | 814 | 10253 |
| Nearby Amenities and Food Options | 729 | 7900 |
| People and Personal Interactions | 943 | 7443 |
| Transportation | 325 | 4073 |
| Value and Pricing | 816 | 9943 |

*Fig 3. Heatmaps of Sentiments Across Aspects*

**Observation:**

- Aspects like "Accommodation Facilities and Aesthetics" and "Location and Transportation" have predominantly positive sentiment.
- "Issues and Complaints" cluster shows relatively more negative sentiment, pinpointing areas for improvement.

## 7.4 Radar Chart for a Highly Reviewed

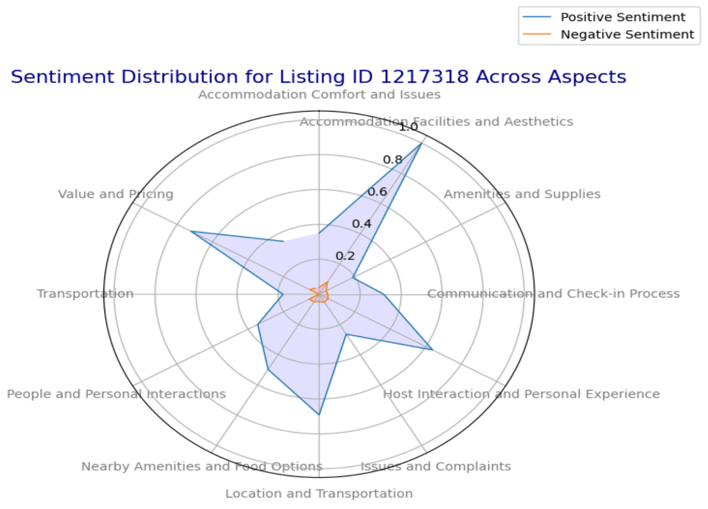


*Fig 4. Sentiment Distribution for Listing ID 1217318 Across Aspects*

**Observation:**

- For the most-reviewed listing, aspects like host interaction and cleanliness received high positive sentiment scores.
- These insights help hosts identify their strengths and understand what consistently pleases guests.

## 8 Summary of Model Performance

**BERT Fine-Tuning Results:**

- Epoch 1: Accuracy ~95%, F1 Score ~97.4%.
- Epoch 2: Accuracy ~95.6%, F1 Score ~97.6%.

These results demonstrate that the fine-tuned BERT model excels in binary sentiment classification.

**Observation:**

- The high F1 score and accuracy indicate the model is well-balanced and effectively identifies both positive and negative sentiments.
- Consistent metrics across epochs reflect minimal overfitting and robust model generalization.

## 9 Summarization (Phase 3) Overview

While my teammate handled most of the work, I contributed by suggesting improvements for prompt engineering and chunking strategies. The T5 model effectively condenses multiple reviews into a cohesive summary, allowing users to quickly understand host qualities, room features, and neighborhood attributes.

## 10 Lessons Learned and Future Improvements

**Key Learnings:**

- **Importance of Preprocessing:** Proper text normalization is crucial for reliable model performance.
- **Benefit of Pre-trained Models:** Using a pre-trained sentiment model jump-started the labeling process.
- **ABSA for Granular Insights:** Aspect-based sentiment goes beyond positive/negative labels, pinpointing what guests specifically like or dislike.
- **Summarization for Decision-Making:** Automated summaries greatly reduce the cognitive load, enabling quick review comprehension.

**Future Directions:**

- Integrate more sophisticated aspect extraction methods like BERTopic or advanced neural topic modeling.
- Address data imbalance through class weighting or oversampling for negative sentiment.
- Explore multi-aspect summarization techniques to highlight different angles of the listing experience.
- Fake Review Detection using Isolation Forest
- Topic Modeling

## 11 Code Provenance and Percentage from External Sources

**Code Sources:**

- Hugging Face Transformers documentation and examples for sentiment analysis, trainer setup, and summarization.
- NLTK and Word2Vec standard methods from official documentation.

**Approximate Percentage from Internet:**

- Lines from Hugging Face and NLTK examples: ~70 lines total
- Modified lines: ~20 lines
- New lines of code: ~ 200 lines
- Percentage:~ 20

The majority of borrowed code relates to model initialization, tokenization examples, and trainer boilerplate. The project-specific logic (data preprocessing steps, clustering, and visualization code) was original.

## 12 References

- Hugging Face Transformers Documentation: https://huggingface.co/docs/transformers
- NLTK Documentation: https://www.nltk.org/
- Kaggle NYC Airbnb Reviews Dataset (2021)
- Contractions Python Library: https://pypi.org/project/contractions/
- Langdetect: https://pypi.org/project/langdetect/
- Word2Vec: Mikolov et al. (2013), "Efficient Estimation of Word Representations in Vector Space"

## 13 Conclusion

My main contribution involved establishing a robust preprocessing and sentiment analysis pipeline. This foundational work enabled the successful extraction of aspects. The comprehensive approach—from raw text cleaning to deep model fine-tuning, aspect-level insights, and final summarization—demonstrates how advanced NLP methods can turn unstructured text into meaningful, actionable insights. Ultimately, this enriches the decision-making process for both potential guests and Airbnb hosts.