Chat Smart on MyLivingCity: Revamping Community Conversations

# Background

The widespread adoption of social media has significantly impacted how communities contribute to decision-making processes. However, the data coming into these systems is often completely unstructured, posing challenges in converting community discussions into productive outcomes. This often leads to repetitive and unproductive conversations. Our project aims to address two main concerns:

* Ensuring discussions remain on-topic and respectful, necessitating a comment filter to prevent inappropriate language.
* Measuring community opinion effectively, as multiple comments may contain similar information, and user engagement can vary widely.

By introducing features like the Comment Funnel Function and advanced filtering systems, we strive to categorize similar comments together, providing users with additional insights and enabling more informed decision-making. Our goal is to facilitate effortless user engagement, allowing users to easily express their thoughts by liking or voting on existing comments, rather than duplicating them on the platform.

# Objectives

In this section, we will highlight the objectives of the Capstone project, including the development of key features aimed at enhancing community interaction and feedback analysis.

## Comment Funnel Function

The objective is to develop and integrate a feature that suggests similar existing comments to users during the submission process. This reduces duplicates and enhances discussion clarity. By encouraging users to agree with existing comments, we streamline the conversation and highlight community consensus. For example, when a user submits a new comment about a local issue, the system will suggest similar existing comments, prompting the user to support or add to those comments instead.

## Input Filtering System

The goal is to design and implement a robust system that detects and prevents inappropriate comments using advanced text analysis techniques. This system will prompt users to modify offensive or off-topic inputs, ensuring constructive and relevant dialogue. For instance, if a user tries to submit a comment with offensive language, the system will detect it and ask the user to rephrase their comment before submission.

## Gauge Statistics and Display Dashboard

We aim to create a comprehensive dashboard to display metrics and statistics from community interactions, providing insights into user segments' agreement and disagreement levels. This dashboard will help users and administrators better understand community dynamics. For example, the dashboard will show the number of comments on a particular topic and the overall sentiment, helping administrators gauge community interest and sentiment trends.

## Summation Comment Display

The objective is to enhance the comment display to present comments in a more structured and visually engaging manner, potentially using techniques to represent community sentiment and feedback. This will help users quickly grasp the essence of the discussions. For instance, a word cloud generated from the comments can visually highlight the most frequently mentioned topics and sentiments.

# Solutions

## Comment Funnel Function

The Comment Funnel Function was developed using sentiment analysis, which employs natural language processing techniques to determine the emotional tone of text. We used a powerful pre-trained large language model to break down each comment into keywords, attitudes, and entities, facilitating the matching of similar comments based on their semantic meaning. When a new comment is submitted, the system compares it to existing comments in the database and suggests similar ones, reducing duplicates and enhancing discussion clarity.

Packages used: NLTK, and transformers (for pre-trained language models).

## Input Filtering System

The Input Filtering System was created using a Python package called "profanity-check," which employs the bag-of-words method to convert text into a numerical representation for efficient processing and analysis. A linear SVM model, trained on a dataset of labeled examples, detects inappropriate comments, including offensive language and off-topic remarks. The system operates at a sub-ms (millisecond) level, ensuring real-time detection and prevention of harmful language.

Packages used: profanity-check, scikit-learn.

## Gauge Statistics and Display Dashboard

The Gauge Statistics and Display Dashboard was developed through the application of sentiment analysis and public user information. This feature analyzes community interactions, including comments, likes, and shares, and visualizes the data through interactive charts and a word cloud, highlighting trends and patterns in community sentiment. This enables users and administrators to better understand community sentiments and make informed decisions.

Packages used: dash, plotly, pandas, WordCloud.

## Summation Comment Display

The Summation Comment Display was enhanced by generating positive and negative sentiment word clouds and developing an automated summary function using the Llama3 model. This function condenses the entire discussion under each topic into a concise summary, providing a comprehensive overview of the community's feedback and sentiments.

Packages used: WordCloud, transformers (for LLaMA-3 model).

# Reflection

As a data science project with a focus on computational linguistics, we had the opportunity to apply our knowledge and skills to develop innovative solutions for MyLivingCity. Through this project, we gained valuable hands-on experience in natural language processing, sentiment analysis, and machine learning.

One key takeaway was the importance of effective feature engineering in developing accurate machine learning models. The Comment Funnel Function showed us how to leverage pre-trained language models to extract meaningful features from text data. Similarly, the Input Filtering System demonstrated the value of converting text into numerical representations using the bag-of-words method for efficient processing and analysis.

We also learned the value of simplicity in model design. The Input Filtering System's linear SVM model proved that a straightforward approach can be both effective and efficient, teaching us to balance model complexity with performance and interpretability.

This project highlighted the impact of data visualization and summarization on community engagement and decision-making. The Gauge Statistics and Display Dashboard and Summation Comment Display features illustrated how condensing complex data into intuitive visualizations can facilitate better understanding and informed decision-making.

Overall, this project allowed us to integrate our knowledge of computational linguistics and data science to drive meaningful outcomes. We are excited to apply these lessons to future projects and continue exploring the intersection of language, data, and decision-making.

# Data

This project is broken down into three sub-tasks, each requiring its own dataset. In this section, we describe the data used for each sub-task, including the source, size, preprocessing steps, and how the data was split for training, development, and testing.

## Comment Funnel Function

### Description:

For the Comment Funnel Function, we utilized YouTube comments data, which shares a similar structure with the MyLivingCity website. Both platforms feature comments organized under different topics (video IDs for YouTube and discussion topics for MyLivingCity), giving suitable samples for us to train our model in summarizing and categorizing. The diverse expression styles in YouTube comments proved beneficial for sentiment analysis and model training.

### Data Source:

**Source:** YouTube comments

**Number of Instances:**

718,745 Rows of GB\_Comments

691,723 Rows of US\_Comments

**Pre Processing:**

It is unrealistic and impractical to manually label every piece of comment data. Therefore, we selected about 200 pieces of data for manual standardization and summarized the keywords and attitudes of the comments. These data served as a benchmark when we trained the model.

**Post Processing:**

After using Meta-LLaMA3 to summarize and refine comments, we manually evaluated approximately 200 randomly selected comments to ensure output met standards and needs. However, we found that the model struggled with accurately capturing tone, attitude, and sentence subjects/objects, leading to adjustments in prompt word engineering. Moreover, some comments contained prohibited words, triggering the model's checking mechanism and preventing accurate summarization. To address this, we extracted these instances, removed the prohibited words, and re-inputted them into the model to generate revised summaries.

## Input Filtering System

### Description:

To develop the Input Filtering System, we employed two datasets: one from Twitter and one from Wikipedia. Both datasets contain human-labeled text samples.

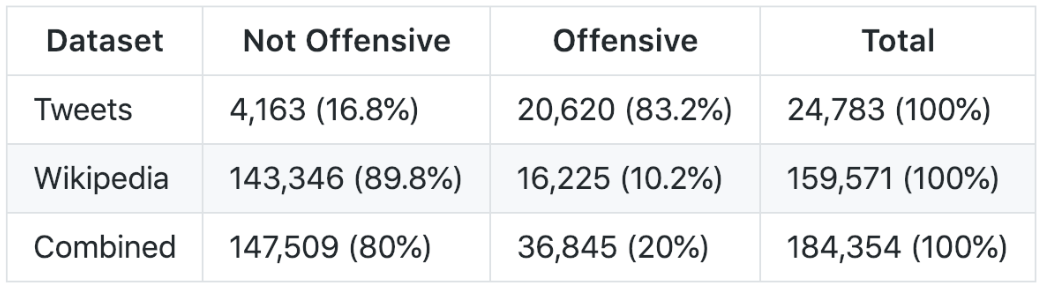
### Data Sources:

**Source:**

[twitter dataset](https://github.com/t-davidson/hate-speech-and-offensive-language/tree/master/data);

[wikipedia comments dataset](https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data)

**Number of Instances:**



**Preprocessing:**

Unify the format of the two datasets;

Vectorize texts using the Bag of Words(BOW) method

## Gauge Statistics and Display Dashboard & Summation Comment Display

### Description:

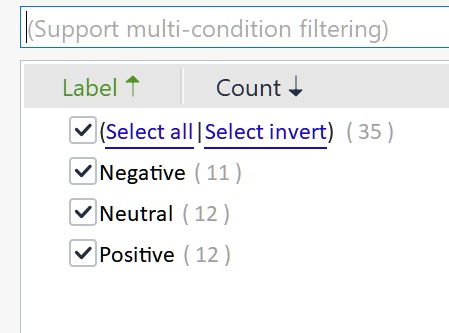
As MyLivingCity is still in development and not publicly accessible, we generated demo data aligned with the website's design structure. This demo data enables the generation of various plots for the Gauge Statistics and Display Dashboard and Summation Comment Display features.

### Data Source:

**Source:** Generated demo data

**Number of Instances:**

There are 35 records in the generated demo data in total with 11 records of Negative tone, 12 records of Neutral, and 12 of Positive tone.



# Methods

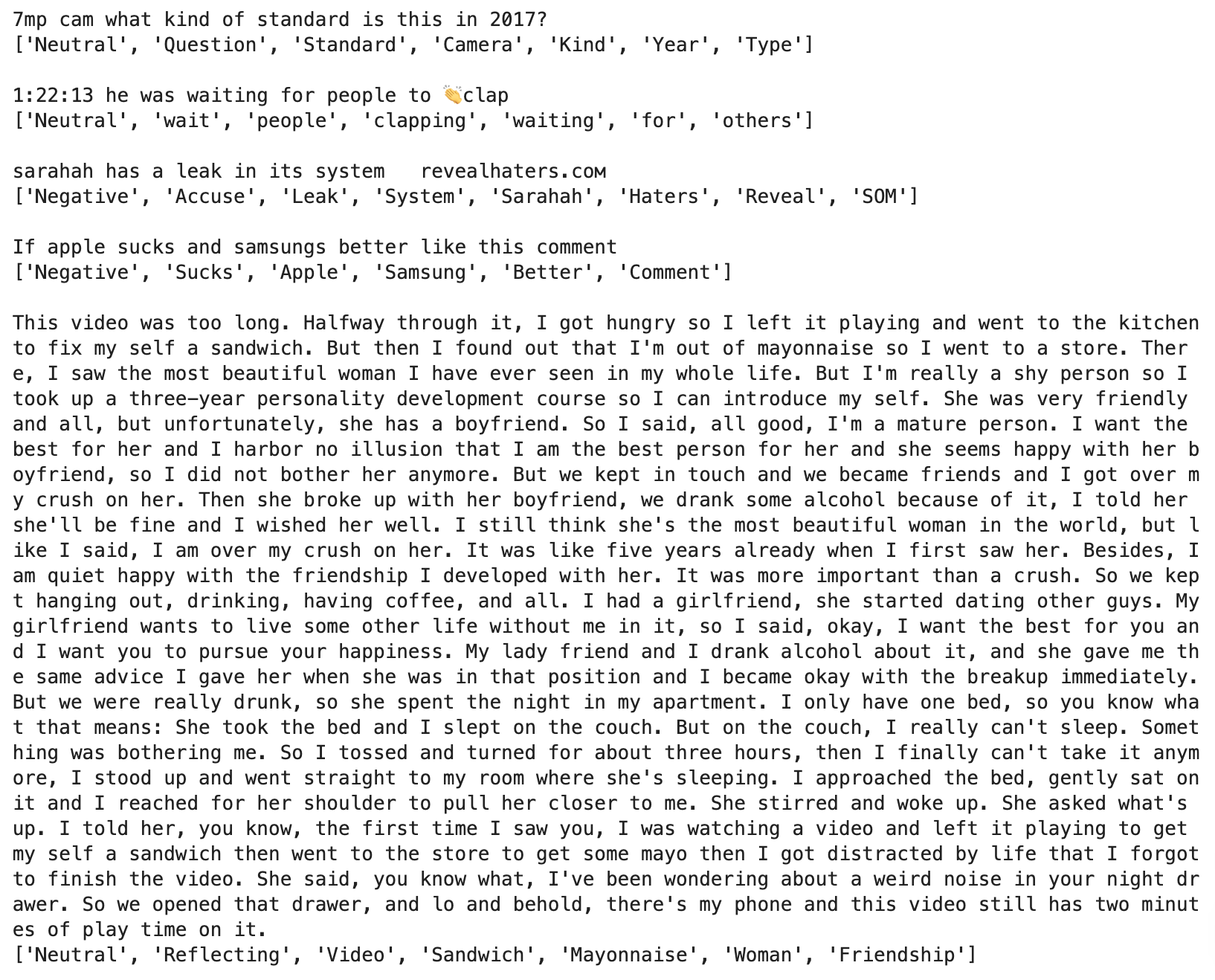
### Comment Funnel Function

To start with the task, each comment in the training dataset was first broken down into individual words or tokens, a process known as tokenization, which helps in analyzing the text at a granular level. After tokenization, we applied lemmatization to convert words to their base or root form. For instance, "running" becomes "run." This step ensures that different forms of a word are treated as the same entity, thereby improving the accuracy of our analysis. Additionally, we removed common stopwords, such as "the," "is," and "in," which do not contribute significant meaning to the text analysis. This helps focus on the more meaningful words in each comment.

To determine the emotional tone of each comment, we employed the Sentiment Intensity Analyzer (VADER) from the Natural Language Toolkit (nltk) library. This tool provides sentiment scores that classify comments as positive, neutral, or negative.

In our pursuit of enhancing comment analysis, we leveraged the capabilities of the powerful pre-trained large language model, Meta-LLaMA3. This model proved instrumental in identifying significant words or phrases that encapsulate the main ideas of each comment, understanding the underlying attitudes expressed, and recognizing entities such as people, organizations, and locations. We were astonished by LLaMA3's impressive performance and summarization abilities. However, we soon discovered that when the prompt words failed to accurately convey our desired outcomes, LLaMA3's output was overly general and occasionally overlooked crucial aspects, such as the commenter's attitude, or confused the subject and object of the comment.

To address these limitations, we employed prompt engineering techniques, refining the prompt words through multiple iterations. Additionally, we utilized ChatGPT4 to score the keywords summarized by LLaMA3 in bulk. Through persistent effort, we ultimately achieved highly satisfactory results. By applying these techniques, each comment was tagged with relevant keywords and attitudes, enabling the creation of a rich, searchable database of comments. When a new comment is submitted, our system compares it to the existing comments in the database based on sentiment and semantic content, including keywords and attitudes. This comparison allows our system to suggest relevant comments, reducing duplicates and enhancing the clarity of discussions.



### Comment Filtering Function

The Input Filtering System was developed using the “profanity-check” Python package, which is specifically designed to detect inappropriate language. The package employs a pre-labeled dataset of text samples, including examples of both offensive and non-offensive language, to train the detection model.

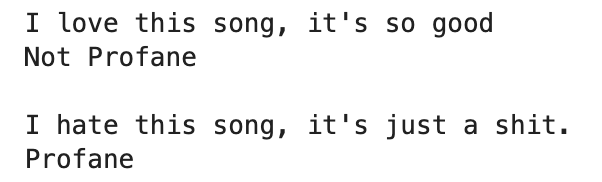
To process the text data efficiently, it used the bag-of-words method. This technique converts text into a numerical representation by creating a matrix of word occurrences, where each row corresponds to a document (in this case, a comment), and each column corresponds to a unique word in the dataset. This representation allows computers to process and analyze textual information more efficiently.

The core of the Input Filtering System is a linear Support Vector Machine (SVM) model. This type of machine learning algorithm is well-suited for classification tasks and is particularly effective in high-dimensional spaces, which is common in text data. The linear SVM was trained using the labeled dataset from the “profanity-check” package. This dataset includes numerous examples of both appropriate and inappropriate comments, enabling the model to learn to distinguish between the two effectively.

Once trained, the linear SVM model can quickly classify new comments as appropriate or inappropriate based on their numerical representation. The system's processing time is extremely fast, operating at a sub-millisecond level, which allows for real-time detection and prevention of harmful language. This speed is crucial for maintaining the flow of conversation and ensuring that inappropriate comments are filtered out promptly.

The Input Filtering System’s implementation is intentionally straightforward to ensure reliability and efficiency. By using the “profanity-check” package, which combines the bag-of-words method with a linear SVM model, the system achieves high accuracy in detecting offensive language and off-topic remarks. This simplicity does not compromise performance; instead, it ensures that the system can operate in real time, maintaining a respectful community dialogue without introducing latency.

Additionally, we explored the potential of utilizing LLaMA3 as a latent language filter to detect novel words derived from prohibited terms. However, due to the strict content moderation mechanisms inherent in large language models and their relatively slow response times, we ultimately abandoned this approach. While LLaMA3's capabilities in generating human-like text are impressive, its limitations in real-time processing and content restrictions hindered its effectiveness in our specific use case. Nevertheless, the Input Filtering System's design, leveraging the "profanity-check" package and linear SVM model, ensures efficient and accurate detection of harmful language, maintaining a safe and respectful community dialogue.



### Gauge Statistics and Display Dashboard

Our project involves the creation of an interactive dashboard using the Dash framework in Python. This dashboard is designed to visualize various aspects of community interaction data, providing insights into user sentiment, user type distribution, gender distribution, income level distribution, age group distribution, and racial/cultural affiliation.

Data Loading and Preprocessing

The dashboard begins by loading a CSV file containing the data using the pandas library. The data is then preprocessed to count the occurrences of different tones, user types, genders, income levels, age groups, and racial/cultural affiliations.

Word Cloud Generation

A word cloud is generated to visually represent the frequency of keywords in the dataset. The WordCloud library is used to create this visualization. The generated word cloud image is converted to a base64 string to be displayed in the dashboard.

Dashboard Layout

The dashboard layout is defined using the Dash framework with dash\_bootstrap\_components for styling. The layout consists of several rows and columns, each containing different interactive elements such as buttons and graphs. The main sections of the layout are:

Title Row: Displays the title of the dashboard.

Button Row: Contains buttons to display various charts.

Chart Rows: Each row contains two columns displaying different charts or visualizations.

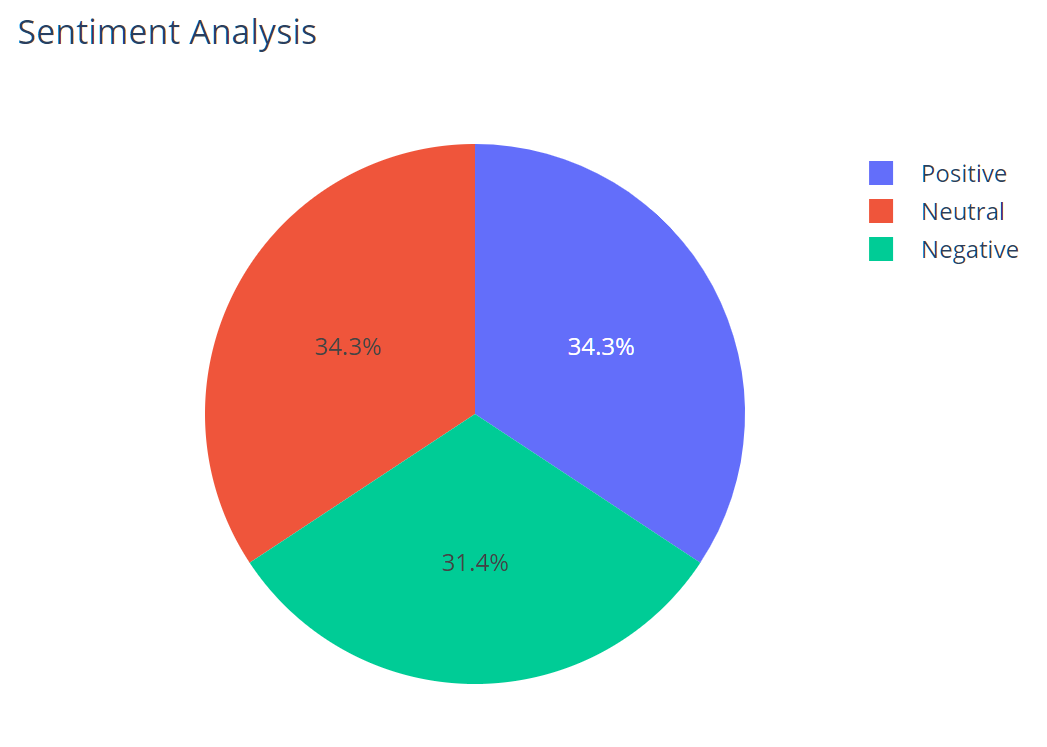
Interactive Elements

The dashboard includes several buttons, each corresponding to a different type of chart. These buttons allow users to toggle the visibility of various charts, making the dashboard interactive and user-friendly.

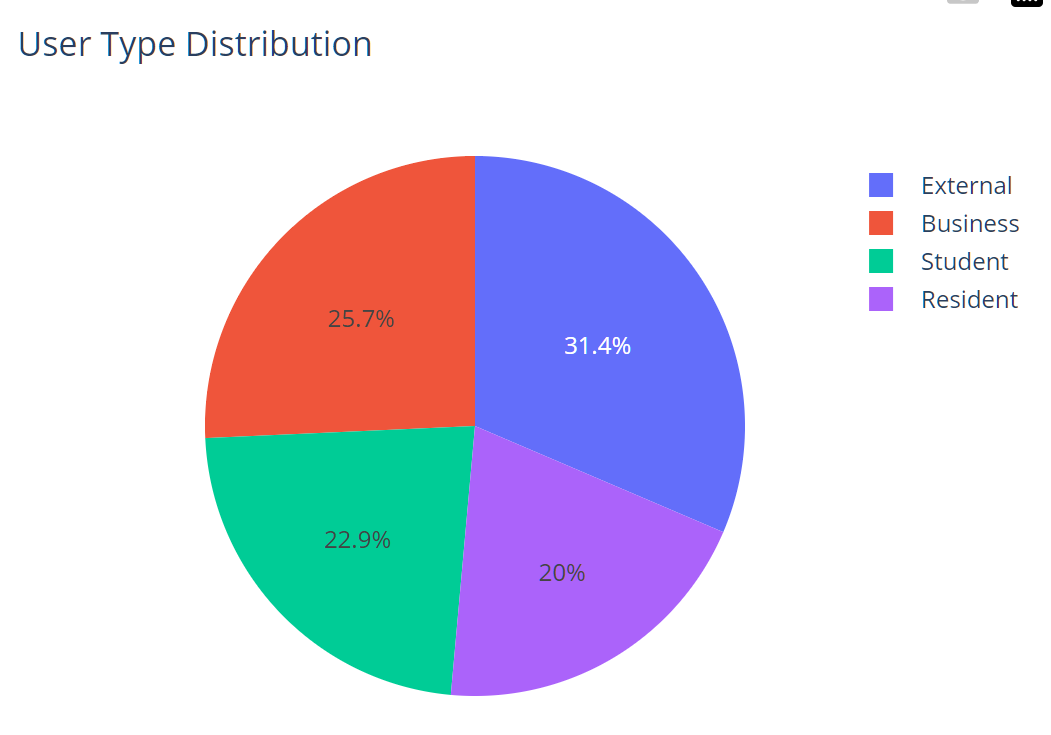
Chart Types

The following types of charts are included in the dashboard:

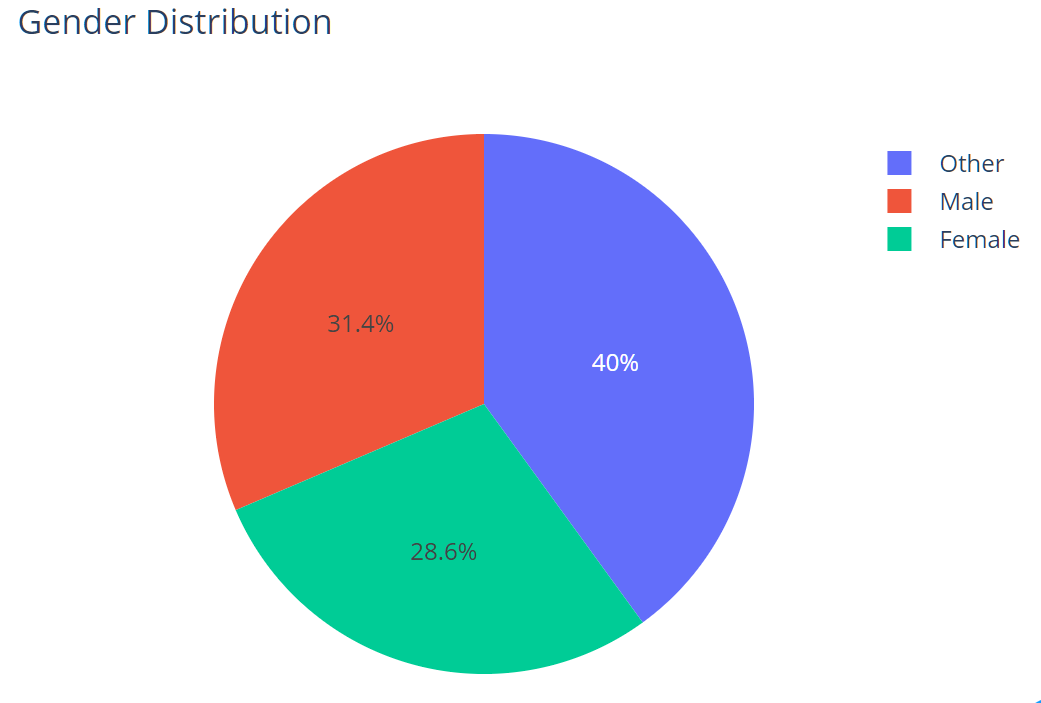
**Sentiment Analysis (Tone) Pie Chart**: Displays the distribution of different sentiment tones in the dataset.



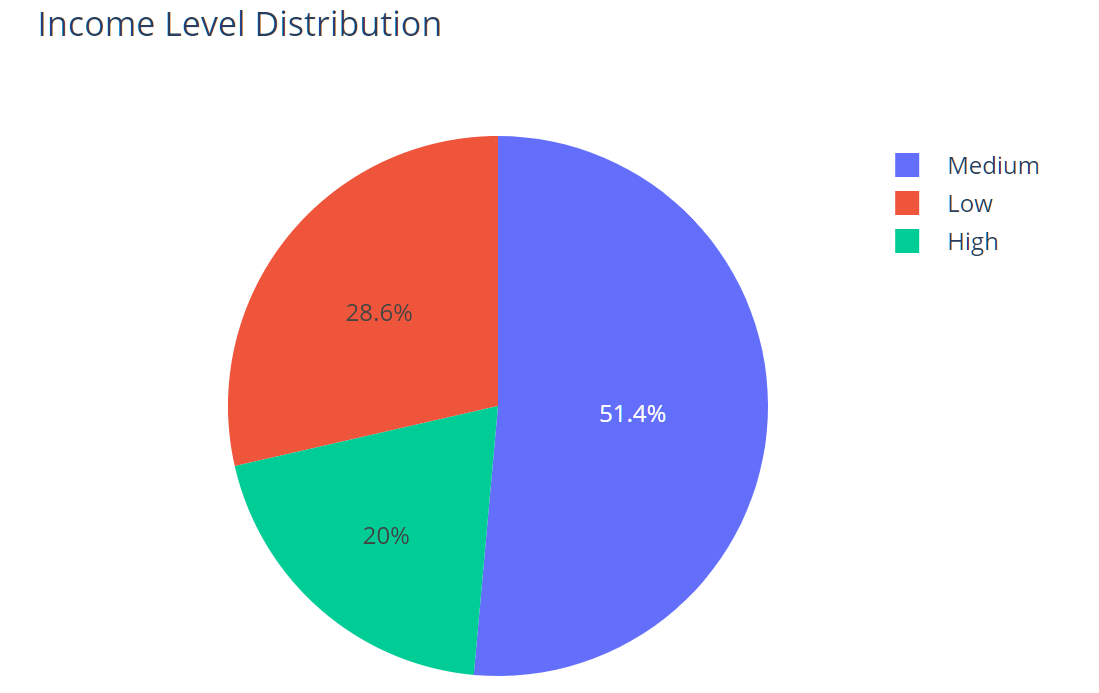
**User Type Distribution Pie Chart**: Shows the distribution of different user types.



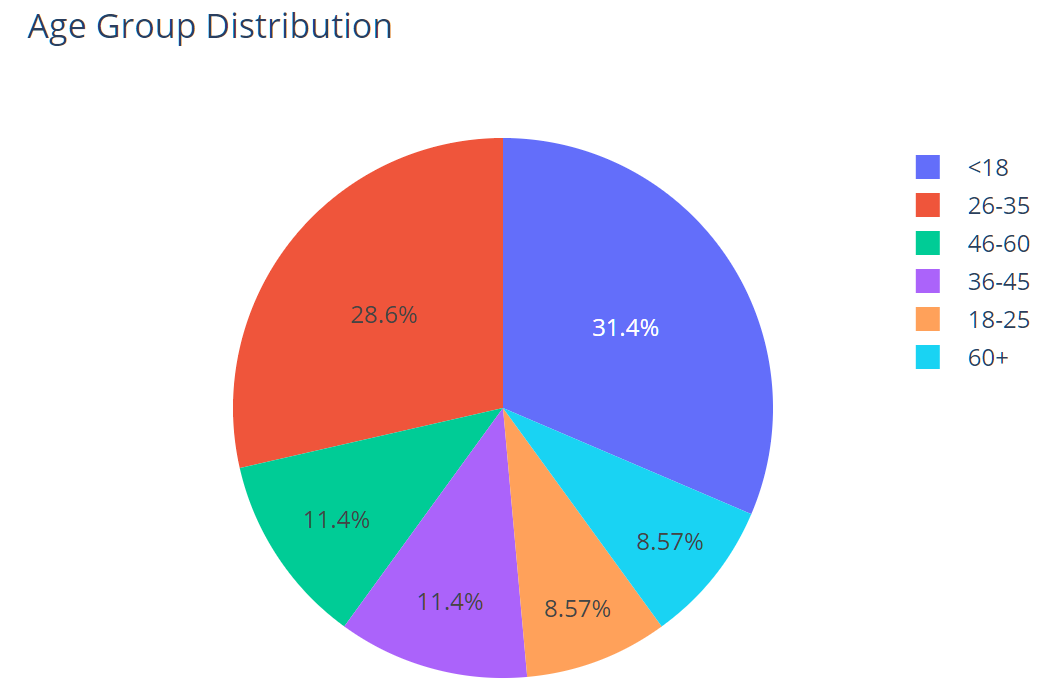
**Gender Distribution Pie Chart**: Visualizes the gender distribution of users.



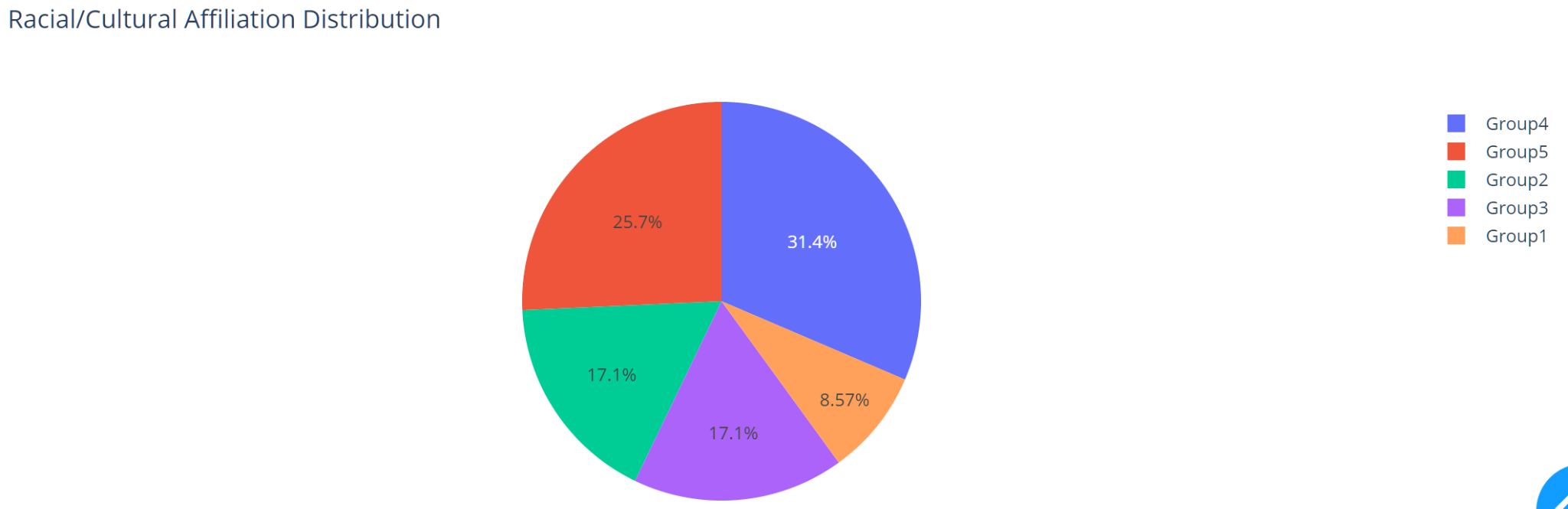
**Income Level Distribution Pie Chart**: Illustrates the distribution of different income levels among users.



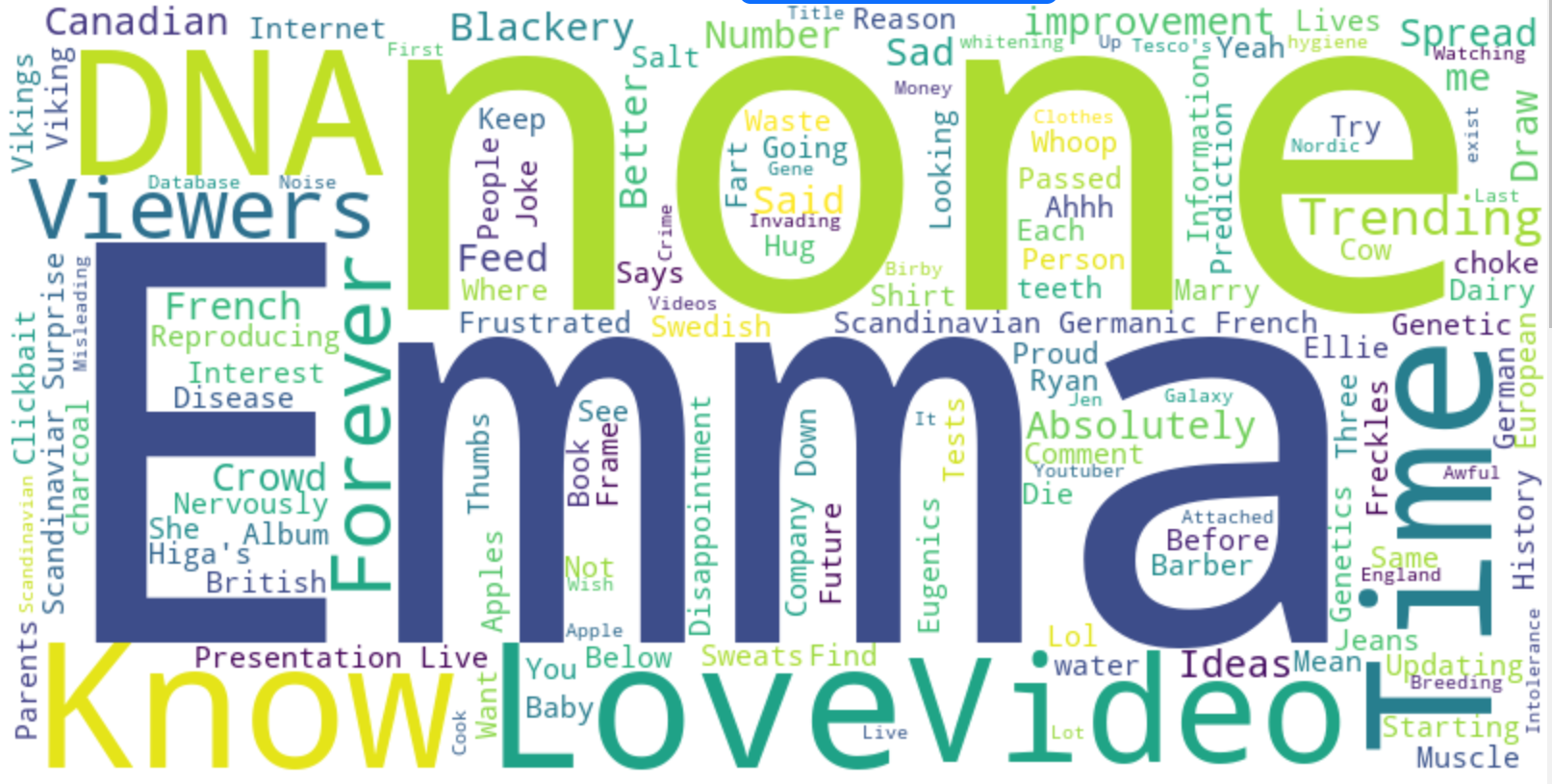
**Age Group Distribution Pie Chart**: Represents the age group distribution of users.



**Racial/Cultural Affiliation Distribution Pie Chart**: Displays the distribution of users' racial and cultural affiliations.



**Word Cloud**: Provides a visual representation of the frequency of keywords in the dataset.



### Summation Comment Display

To develop the Summation Comment Display, we began by performing sentiment analysis on the comments. This step involved identifying and categorizing the emotional tone of each comment as positive or negative. Using the results of this sentiment analysis, we created two separate word clouds: one for positive sentiment words and another for negative sentiment words. These word clouds visually highlight the most frequently used words and phrases within the comments, providing users with a quick and intuitive understanding of the dominant sentiments in the discussion.

The core of the Summation Comment Display is the automated summary function, which was developed using the Llama3 model, a powerful pre-trained large language model. To generate concise summaries of the discussions under each topic, we employed prompt engineering techniques with Llama3. This involves crafting specific prompts to guide the model in extracting the most relevant and significant points from the entire set of comments.

The Summation Comment Display's automated summary function, powered by LLaMA3, enables the rapid generation of concise summaries that correspond to a series of comments. Leveraging LLaMA3's impressive processing capabilities and generous token handling, we can input over 100 comments at once, allowing the model to produce a detailed summary and extract key keywords. In practice, the output results have been exceedingly satisfactory, with the model accurately summarizing the mainstream opinions in the comments and even capturing minor attitudes that diverge from the mainstream views.

# Evaluation

### Comment Funnel Function

To evaluate the summaries generated by LLaMA3 for each comment, we employ a hybrid approach that combines human assessment with another large language model. This method is necessitated by the absence of a universal evaluation metric for summarizing unstructured text, where no standardized benchmarks exist. After summarizing and extracting keywords from a particular comment, we conduct manual reviews and utilize ChatGPT to inversely generate the original text based on the keywords, then assess the semantic similarity between the inferred original text and the source comment. This approach enables us to evaluate the fidelity and accuracy of the summaries generated by LLaMA3.

Supplementally, through this method, we found that LLaMA3 was able to correctly infer text similar in meaning to the original text for the vast majority of comments, with the exception of a small number of extremely brief and ambiguous comments. Therefore, we believe that this function has met our expectations.

### Comment Filtering Function

Regarding the comment filter function, based on our standardized dataset, we employed metrics such as accuracy and F1 score to evaluate its performance. Our findings indicate that your model exhibits robust performance, achieving a test accuracy of 95.0% and a balanced test accuracy of 93.0%. The precision stands at 86.1%, and the recall at 89.6%, yielding an F1 score of 0.88. This suggests that your model demonstrates both accuracy and balance in handling diverse classes, ensuring reliable predictions overall. These results are gratifying, leading us to believe that this function can effectively filter out prohibited words in comments in most scenarios.

### Dashboard

In developing the Gauge Statistics and Display Dashboard, we prioritized data integrity, user-friendliness, and insightful visualizations. Using pandas for data preprocessing ensured efficient manipulation and integrity, while the WordCloud library provided clear keyword frequency visualization. The dashboard, built with Dash and dash\_bootstrap\_components, emphasized responsiveness and aesthetic appeal, and tested across devices for user feedback. Interactive elements were implemented with Dash’s callback functionality, ensuring real-time updates and correct chart control, and thoroughly tested for immediate responsiveness. We used Plotly Express to create clear, informative pie charts for user type, gender, income level, age group, and racial/cultural affiliation distributions, and a bar chart for sentiment analysis, ensuring clear labeling and valuable insights. Overall, the iterative testing and feedback refined the dashboard, ensuring a seamless, interactive, and insightful user experience.

### Summation Comment Display

For the Summation Comment Display feature, due to its unique nature, we lacked a "gold standard" to assess the adequacy of the summary. After careful consideration, we resorted to manual evaluation, where we scrutinized the summaries generated for a large number of comments on the same topic and compared them to the topic's corresponding word cloud. Additionally, we solicited feedback from peers on the summary text. To a certain extent, we deemed the summary to be in line with our requirements, meeting our needs satisfactorily.

# Analysis

### Comment Funnel Function

Regarding this functionality, we acknowledge room for improvement. Constrained by server performance, we cannot employ a pairwise comparison approach to select the most similar comments. Instead, we resort to summarizing keywords and comparing them to rapidly filter out potentially similar content. During testing, we observed that if the source database contains limited data, it may fail to retrieve similar comments. In future development, we consider summarizing text information into vectors, which can mitigate potential matching issues when comparing keywords to a certain extent. This approach may enhance the efficiency and accuracy of our similarity detection algorithm.

### Comment Filtering Function

We are exceedingly satisfied with the performance of the Comment Filtering Function, which leverages a relatively simple SVM algorithm, thereby yielding a significantly faster response time compared to LLM. Moreover, it demonstrates remarkable proficiency in tackling the vast majority of profanity-filtering tasks, leading us to conclude that this function has met our requirements. Its efficiency and accuracy in classifying and filtering comments have exceeded our expectations, making it a valuable asset in maintaining a safe and respectful online environment.

### Dashboard

From a user perspective, the redesigned dashboard offers several significant benefits. The interactive elements and clear visualizations make it easier to explore and understand complex data. By choosing a word cloud for keyword frequency, we prioritized a visually appealing method to quickly grasp the most prominent keywords, enhancing the user's ability to identify key trends and topics at a glance. The sentiment analysis presented in a pie chart provides an intuitive overview of data distribution, making it easy to identify dominant sentiments and facilitating quicker decision-making.

The decision to exclude the top 20 attitude distribution and top 50 keyword frequency charts in favor of more holistic and visually appealing representations ensures that the dashboard remains uncluttered and user-friendly. This not only improves the user experience but also ensures that the most critical insights are highlighted without overwhelming the user with too much information.

The potential impacts of these design choices are substantial. Enhanced clarity and usability can lead to better and faster decision-making, as users can more easily interpret the data. This can improve operational efficiency and strategic planning. Additionally, the interactive features encourage more engagement with the data, potentially leading to deeper insights and more innovative solutions.

### Summation Comment Display

Regarding the Summation Comment Display feature, this is the optimal approach we have achieved so far. Leveraging LLaMA3 enables us to summarize and condense original text into a concise summary, approximately one-fifth to one-tenth the length of the original text while maintaining a high level of fidelity. Moreover, these summaries have received unanimous recognition in human evaluations, indicating that they meet our requirements. By utilizing LLaMA3, we have successfully distilled the essence of the original text, preserving the most valuable information in a condensed format, thereby fulfilling our objectives.

# Limitation

While this project showcases the feasibility and potential impact of the proposed solutions, several limitations must be acknowledged due to the nature of the data used and the development context. Specifically, the requirement for substantial memory and GPU resources to run and fine-tune large language models as LLaMA3 constrains us from utilizing third-party hosted LLaMA3 API services instead of local deployment. Currently, the online API services we have found charge $0.08 per million tokens, which, based on our estimates, can summarize approximately 2,000 comments, representing a reasonable and acceptable cost. If demand for the service increases in the future, we may reconsider deploying the large model locally.

### Data Differences:

The principal constraint of this project lies in the fact that the models and solutions were developed and tested on datasets that diverge from the actual live data MyLivingCity will utilize. We employed YouTube comments, Twitter data, and Wikipedia talk page edits as surrogates for the anticipated user interactions on MyLivingCity, acknowledging that these datasets share similarities with the expected MyLivingCity data but may not fully capture the distinctive characteristics and context-specific subtleties of the final user interactions on the platform. Moreover, during the summarization process, discrepancies in verb tense and singular/plural forms of words may emerge, leading to potential difficulties in matching the summarized key words, thereby affecting the accuracy of the model. This limitation highlights the need for further fine-tuning and adaptation of the models on the actual MyLivingCity data to ensure optimal performance.

### Model Generalizability:

Given the different data sources, there is a possibility that the performance of our models might vary when applied to live data. The sentiment analysis, profanity detection, and comment summarization models were all trained and validated on surrogate datasets. While we chose these models and methods as the most appropriate for the problem based on available data, the results may not generalize perfectly to the live MyLivingCity data. Factors such as differing user demographics, language use, and discussion topics could impact model accuracy and effectiveness.

### Demo Data Usage:

For the Gauge Statistics and Display Dashboard and Summation Comment Display features, we generated demo data that mimicked the design structure of MyLivingCity. This synthetic data enabled us to create and test the visualizations and summary functions, but it does not reflect actual user interactions and sentiments. Consequently, the insights derived from these features in a live environment may differ from those observed during the project development phase.

### Model Adaptability:

Although we employed robust preprocessing techniques and advanced models like VADER and Llama3, the adaptability of these models to new, unseen data remains uncertain. The models may require further tuning and retraining once live data is available to ensure optimal performance and relevance.

### Real-time Performance:

The Input Filtering System was designed for real-time performance, operating at a sub-millisecond level. However, the actual performance in a live environment with concurrent users and varying load conditions remains to be tested. Ensuring that the system maintains its speed and accuracy under live conditions will be crucial for its practical implementation.

In summary, while we have chosen and implemented the most suitable models and solutions based on the data and resources available during this project, the transition to live data may present unforeseen challenges. Future work will involve validating and potentially adapting these models with real MyLivingCity user data to ensure their continued effectiveness and relevance in enhancing community interactions and decision-making processes.

# Future Work

One significant area for future work is the integration of adaptive learning mechanisms. Currently, our models are trained on a fixed dataset, which limits their ability to adapt to new language trends and emerging types of inappropriate content. By implementing a continuous learning framework, the system can periodically update its training with new data. This approach will ensure that the models remain current and effective, adapting to the dynamic nature of language and community interactions. Continuous learning will also allow the system to refine its algorithms based on real-time user feedback and behavior, enhancing its accuracy and relevance over time.

Another promising avenue for enhancement is the development of advanced visualization techniques. The current dashboard provides basic metrics and statistics, but there is potential to integrate more sophisticated data visualization tools, such as interactive dashboards, network graphs, and heatmaps. These advanced visualizations can offer deeper insights into community dynamics, sentiment trends, and user interactions. For example, interactive charts could allow users and administrators to explore data in more detail, identifying specific patterns and trends that are not immediately apparent in static displays. This level of detail and interactivity will facilitate better understanding and more informed decision-making, ultimately leading to more effective community engagement.

Improving the overall user experience is also a critical area for future development. Enhancing user experience involves creating a more intuitive, responsive, and personalized interaction environment. One approach is to introduce personalization features that tailor content and suggestions based on individual user preferences and interaction history. By leveraging machine learning algorithms, the system can learn from user behavior to provide more relevant and engaging recommendations. Additionally, refining the user interface to make it more user-friendly and accessible will encourage more active participation from the community. This focus on user experience will not only increase engagement but also ensure that users feel valued and understood, fostering a stronger sense of community.

In summary, focusing on adaptive learning, advanced visualization techniques, and user experience will significantly enhance the effectiveness and impact of the Comment Funnel Function and Input Filtering System. By ensuring the system remains up-to-date with language trends, providing deeper insights through sophisticated visualizations, and creating a more engaging user experience, we can foster a more informed, inclusive, and dynamic community dialogue.

# 

# Timeline

| Week | Dates | Tasks | Supplementary Tasks |
| --- | --- | --- | --- |
| Week 1 | May 6 - May 12 | * Data Collection and Preprocessing; * Finalize project plan; * Comment Funnel Function development start |  |
| Week 2 | May 13 - May 19 | * Continue Comment Funnel Function development; * Input Filtering System development start |  |
| Week 3 | May 20 - May 26 | * Continue Input Filtering System development | * Integrate functions to website |
| Week 4 | May 27 - June 2 | * Gauge Statistics and Display Dashboard Development | * Integrate functions to website |
| Week 5 | June 2 - June 8 | * Summation Comment Display Development | * Integrate functions to website * Modify dashboard |
| Week 6 | June 9 - June 15 | * Cleaning and standardizing existing codes; * Handover document preparation | * Integrate functions to website |
| Week 7 | June 16 - June 22 | * Final project report preparation; * Presentation preparation | * Integrate functions to website |

# 