

# AD ENGAGEMENT PROJECT REPORT

*IML*



**Harsh Kumar B22ES027**

**Ayush Goel B22ES013**

**Aditya Kumar B22ME004**

**Aryan Sharma B22BB011**

**Sonal Raj B22ME062**

## Problem Statement

Analyze user interactions with sponsored posts and advertisements. Measure the effectiveness of ad campaigns in driving engagement and conversions. Use the following kaggle database -

<https://www.kaggle.com/datasets/sanjanchaudhari/user-behavior-on-instagram>

## Basic Agenda

The primary objective of our project is to rigorously assess the user-generated data encapsulated within the dataset. This involves a comprehensive analysis of user interactions, preferences, and behaviors, extracted from the provided dataset. The pivotal aim is to distill meaningful insights that can inform the design of a robust algorithm. This project, once completed, is anticipated to serve as a catalyst for fostering effective user engagement across diverse ad campaigns.

## Basic Overview of the Database

	Unnamed: 0	id	comment	User id	Photo id	created Timestamp	posted date	emoji used	Hashtags used count
0	0	1	unde at dolore	2	1	13-04-2023 08:04	April 14	yes	1
1	1	2	quae ea ducimus	3	1	13-04-2023 08:04	April 14	no	2
2	2	3	alias a voluptatum	5	1	13-04-2023 08:04	April 14	no	4
3	3	4	facere suscipit sunt	14	1	13-04-2023 08:04	April 14	yes	2
4	4	5	totam eligendi quaerat	17	1	13-04-2023 08:04	April 14	yes	1

Basic Overview of database

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	7488.0	3743.500000	2161.743741	0.0	1871.75	3743.5	5615.25	7487.0
id	7488.0	3744.500000	2161.743741	1.0	1872.75	3744.5	5616.25	7488.0
User id	7488.0	48.949386	28.354045	2.0	24.00	48.0	72.00	100.0
Photo id	7488.0	129.099225	73.776439	1.0	65.00	130.0	193.00	257.0
Hashtags used count	7488.0	2.416667	1.705905	0.0	1.00	2.0	3.25	6.0

Mathematical features of database

Certain useful features include: user id, comment, photo id, hashtags count, emoji used

## What do we derive from this overview?

**Description:** The user database under consideration is a unique and intriguing dataset that captures user engagement and behavior within the Instagram platform. What sets this dataset apart is its distinct characteristic of having comments expressed in the classical language of Latin, adding a cultural and historical nuance to the user interactions. This linguistic feature not only distinguishes it from conventional datasets but also adds an academic and artistic dimension to the study of contemporary social media behavior.

The core components of the dataset include:

1. **Number of Entries (Unnamed: 0):** The dataset comprises a total of 7488 entries, each representing a unique instance of user engagement on Instagram.
2. **User IDs (id and User id):** There are 7488 unique user IDs, indicative of the diverse set of individuals contributing to the dataset. The User id column, with 77 unique values, suggests that some users might have multiple entries, emphasizing the longitudinal aspect of user behavior study.
3. **Comments in Latin (comment):** A vast majority of the dataset, specifically 7467 entries, consists of comments expressed in Latin. This linguistic attribute adds an unconventional and scholarly dimension to the user-generated content, potentially requiring specialized natural language processing techniques for comprehensive analysis.
4. **Photo IDs (Photo id):** The dataset incorporates 257 unique photo IDs, signifying the variety of visual content that users engage with on Instagram.
5. **Temporal Information (created Timestamp and posted date):** The temporal aspect of user engagement is tracked through the created Timestamp and posted date columns. While the former provides a precise timestamp for each interaction, the latter might capture the date of posting with a coarser granularity.
6. **Emoji Usage (emoji used):** Users' use of emojis is a crucial aspect of modern communication. The dataset includes a column indicating the type and frequency of emojis used in the comments, reflecting the emotive and expressive nature of user interactions.
7. **Hashtags Usage (Hashtags used count):** The dataset further explores the use of hashtags in comments, with counts ranging from 0 to 7. This feature sheds light on the extent to which users leverage hashtags for content categorization and discoverability.

```

Unnamed: 0      7488
id              7488
comment        7467
User id         77
Photo id       257
created Timestamp    1
posted date        1
emoji used        2
Hashtags used count  7
dtype: int64

```

No. of unique numerical values in each features

**We imputed certain useless features such as “posted date”, “created timeline” and “id”**

```

[ ] #Dropped unnecessary columns
    df.drop(['id', 'created Timestamp', 'posted date', 'Unnamed: 0'], axis=1, inplace=True)
    df['emoji_used'] = df['emoji used']

```

We used label encoding to convert the emoji used column into binary numerical values 0 and 1.

```

#Label Encoding emoji used column to 1 or 0
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
df['emoji_used'] = label_encoder.fit_transform(df['emoji used'])
df

```

- Label encoding is a common technique used in machine learning to convert categorical data into numerical form, specifically when dealing with algorithms that require numerical input. In this case, you applied label encoding to the 'emoji used' column in your DataFrame using the `LabelEncoder` from the `sklearn.preprocessing` module. The purpose of this transformation is to represent each unique emoji with a corresponding numerical value.

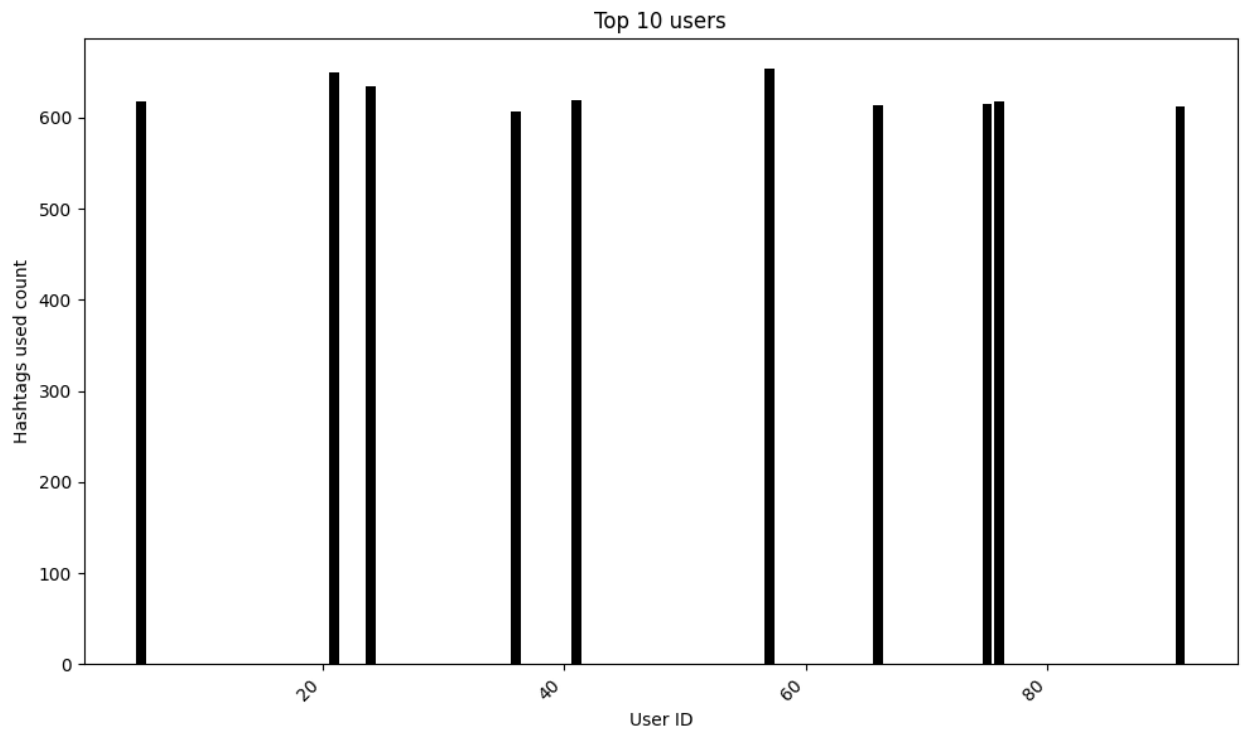
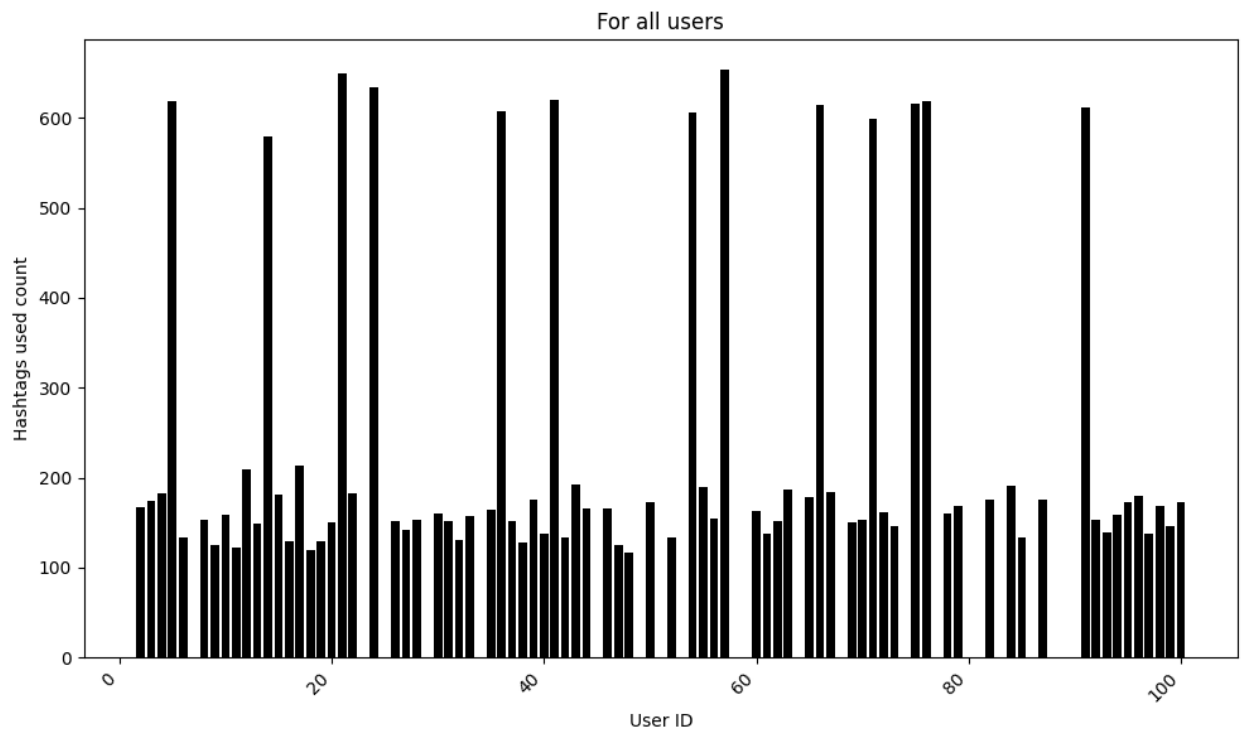
## Exploratory Data Analysis

We employed various EDA strategies to get to the crux of data, That includes and are not limited to

1. **Descriptive Statistics:** Utilizing summary statistics such as mean, median, mode, range, and standard deviation to provide a high-level understanding of the central tendencies and variability in the data.
2. **Data Visualization:** Creating visual representations of the data using charts and graphs, such as histograms, box plots, scatter plots, and pie charts. Visualization helps in identifying patterns, trends, and potential outliers in the dataset.
3. **Correlation Analysis:** Examining the relationships between different variables in the dataset by calculating correlation coefficients. This helps identify potential dependencies and understand how changes in one variable may affect another.
4. **Missing Values Imputation:** Investigating and addressing missing values in the dataset. Strategies might include imputing missing values using mean, median, or more sophisticated techniques based on the nature of the data.
5. **Outlier Detection:** Identifying and handling outliers that might skew the analysis. Techniques such as Z-score analysis or IQR (Interquartile Range) can be employed to detect and address outliers.
6. **Feature Engineering:** Creating new features or transforming existing ones to extract more meaningful information from the data. This could involve creating interaction terms, scaling features, or converting categorical variables into numerical representations.
7. **Distribution Analysis:** Assessing the distribution of variables to understand their shapes and characteristics. This is particularly important in determining whether the data follows a normal distribution or if it exhibits skewness.

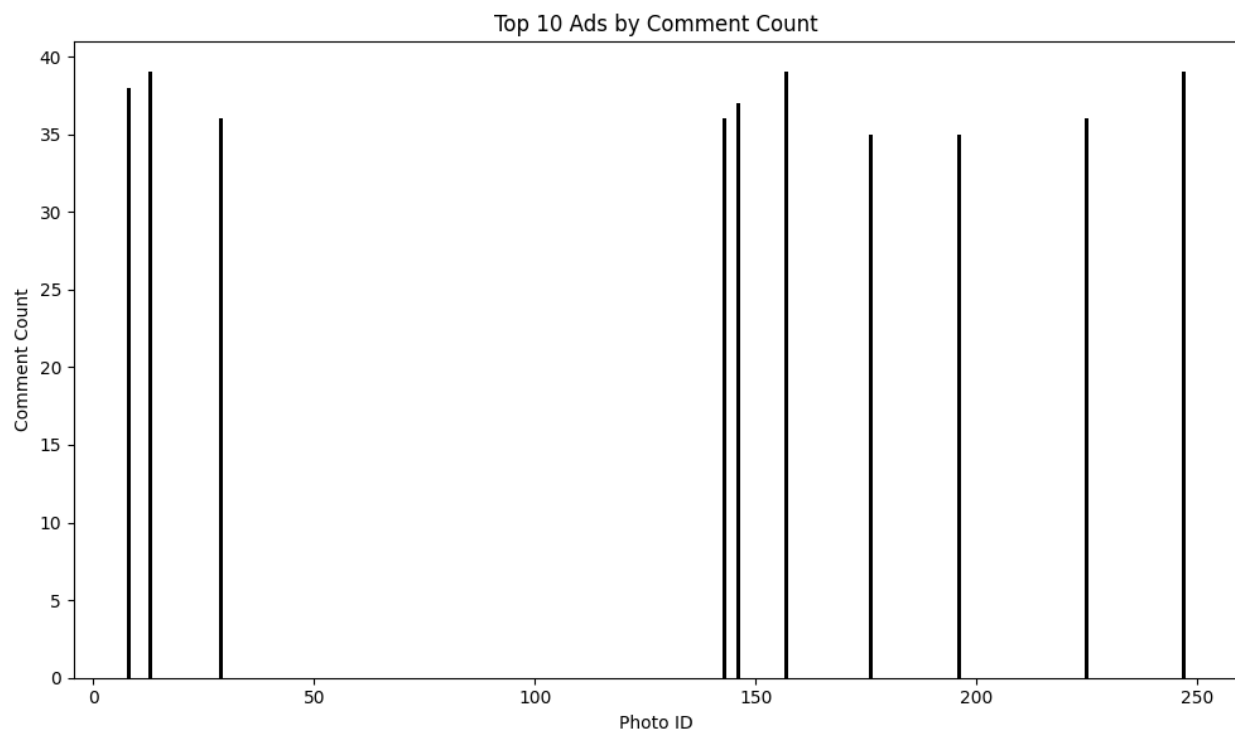
- ☐ Here, we conducted an analysis on the Instagram dataset, specifically focusing on users' hashtag usage. The top 10 users with the highest total count of hashtags were identified and displayed, showcasing their respective 'User id' and 'Hashtags used count.' Additionally, we visualized the overall distribution of hashtag usage across all users using a bar chart, where each bar represented an individual user and its height denoted the total number of hashtags used. Furthermore, we created a separate visualization specifically for the top 10 users, providing a more detailed insight into the hashtag engagement patterns of the most active participants. These analyses and visualizations offer a comprehensive view of how users contribute to the platform through their hashtag usage, highlighting both individual top performers and the broader

distribution trends across the entire data

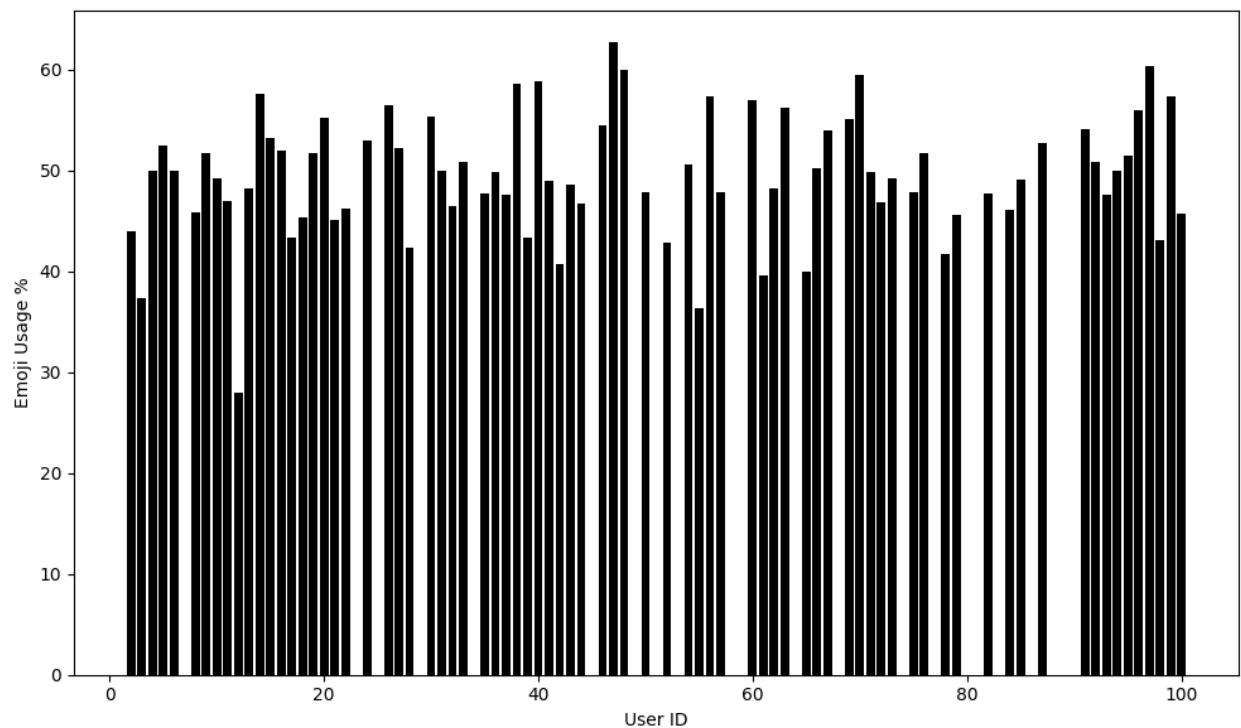


	User	id	Hashtags used	count
46		57		654
18		21		650
20		24		634
34		41		620
60		76		618
3		5		618
59		75		615
52		66		614
67		91		612
29		36		607

In the analysis of Instagram user behavior, the top users with the highest total counts of hashtags are particularly noteworthy. User 57 emerges as the most prolific contributor, employing a remarkable 654 hashtags, closely followed by users 21 and 24 with counts of 650 and 634, respectively. This concentrated group of top users demonstrates a high level of engagement and influence in shaping hashtag trends on the platform. These results underscore the unequal distribution of hashtag usage, highlighting a select group of users who significantly contribute to the overall dynamics of user-generated content on Instagram.

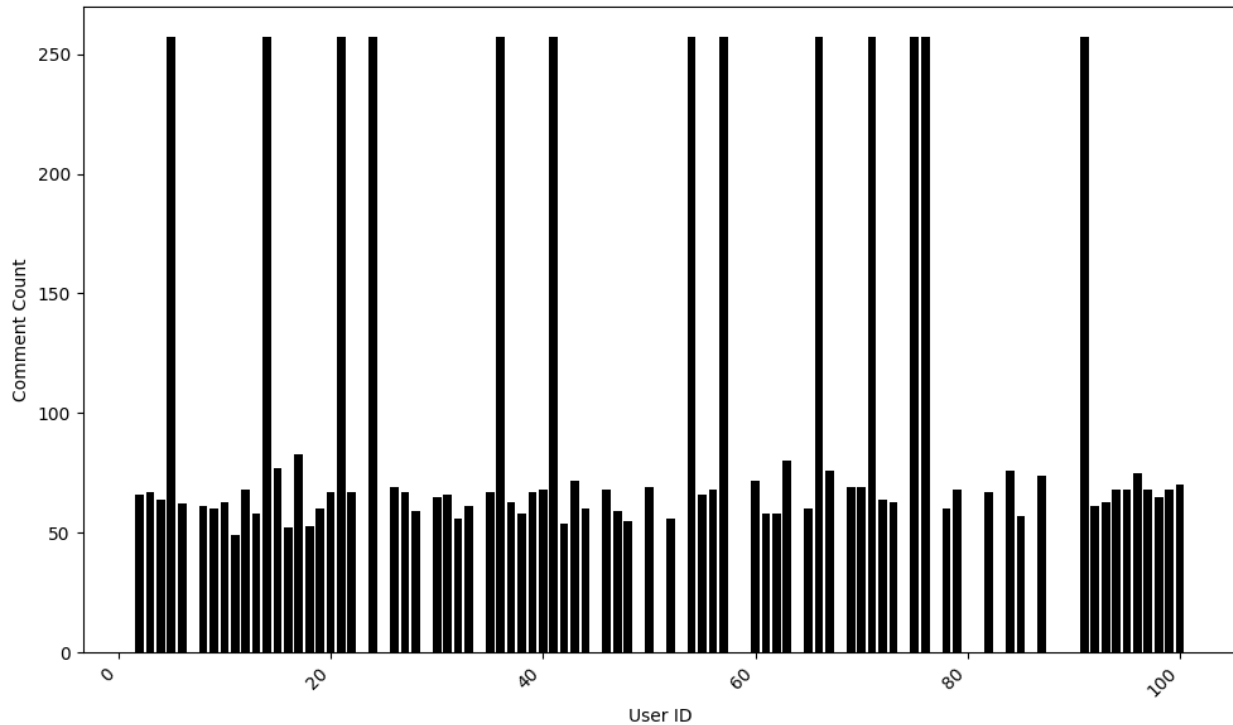


Here, we identified and presented the top 10 photos on Instagram based on the highest comment counts. The results were visualized through a bar chart, where each bar corresponds to a specific photo, and the height of the bar represents the respective comment count. The use of black-colored bars enhances the visibility of the chart, allowing for a clear comparison of comment counts across the top 10 photos. This visualization provides a quick and insightful overview of the most commented-on photos, offering a visual hierarchy of user engagement on the platform.



Here, we computed and showcased the top users on Instagram based on the percentage of comments containing emojis. The findings were presented visually using a bar chart, where each bar represents a specific user, and the bar's height signifies the percentage of their comments that include emojis. The strategic use of black-colored bars enhances the clarity and visibility of the chart, allowing for a straightforward comparison of emoji usage percentages among the top users. This visualization offers a concise and insightful summary of the most emoji-engaged users on the platform, emphasizing the role of certain individuals in contributing to the expressive and emotive content within the Instagram community.





Here, we tabulated and exhibited the comment counts for individual users on Instagram, highlighting the top 20 users with the highest comment counts. The results were further visualized through a bar chart, where each bar corresponds to a specific user, and the height of the bar represents their respective comment count. The deliberate use of black-colored bars enhances the visibility of the chart, ensuring a clear and impactful presentation of user engagement. This visualization provides a concise overview of the most active contributors in terms of commenting, showcasing the significant role played by these top users in fostering interaction and dialogue within the Instagram community.

## Correlation Matrix:

- A correlation matrix is a statistical tool that provides a comprehensive overview of the relationships between variables in a dataset. It calculates correlation coefficients, which quantify the degree and direction of linear relationships between pairs of variables. The correlation coefficient ranges from -1 to 1, where:
  - 1 indicates a perfect positive linear relationship,
  - -1 indicates a perfect negative linear relationship, and
  - 0 indicates no linear relationship.

Here, we generate a square matrix where each cell represents the correlation between two variables. The matrix is symmetric, with the diagonal elements always being 1 since a variable perfectly correlates with itself.

Interpreting the correlation matrix involves examining the magnitude and sign of correlation coefficients. Positive coefficients indicate a positive association, meaning as one variable increases, the other tends to increase as well. Negative coefficients signify a negative association, indicating that as one variable increases, the other tends to decrease.

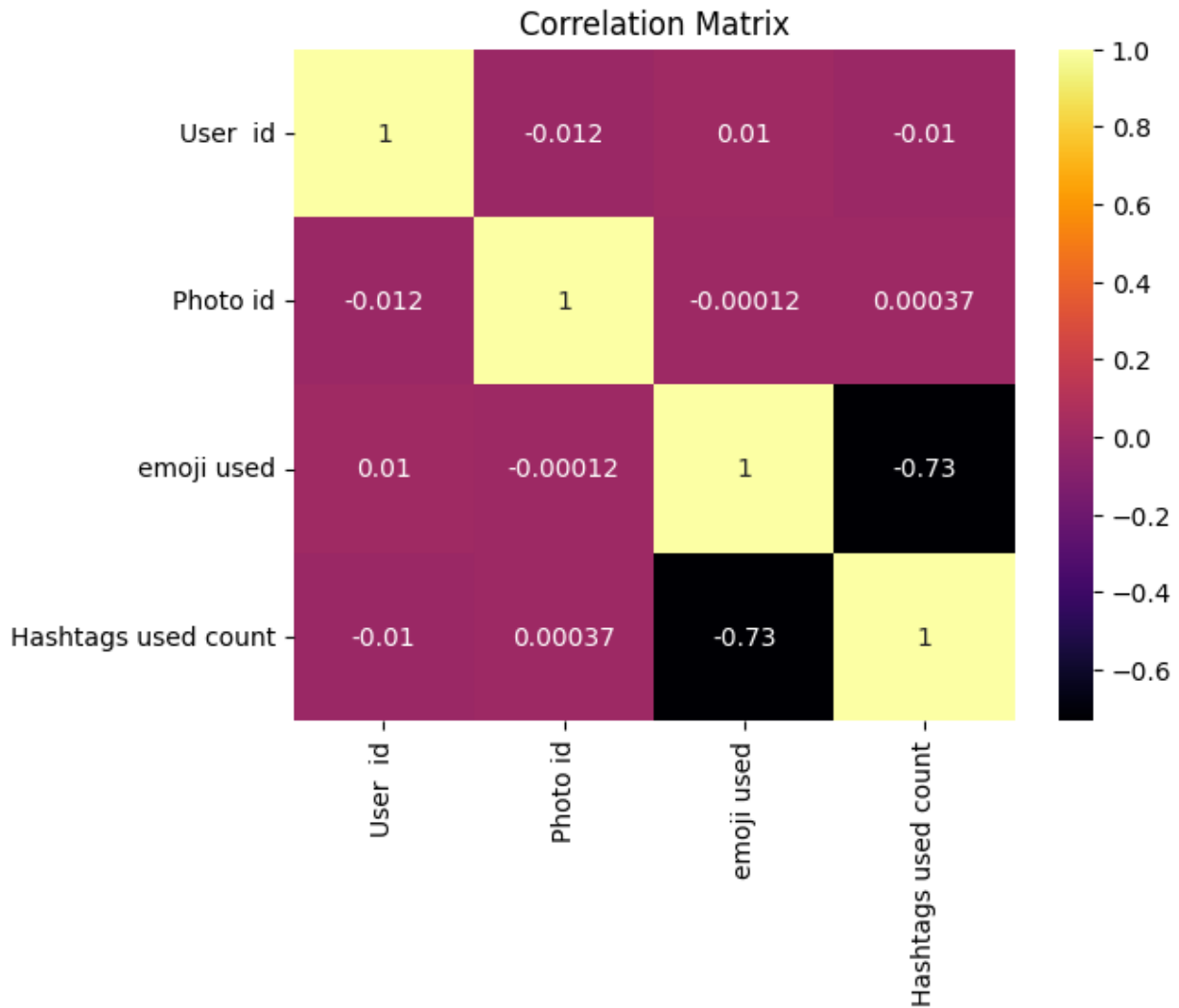
A correlation matrix is a powerful tool for several reasons:

1. **Identifying Relationships:** It helps identify which variables are positively, negatively, or not correlated, offering insights into potential patterns or connections within the data.
2. **Multicollinearity Detection:** It helps detect multicollinearity, which occurs when two or more variables are highly correlated. High multicollinearity can affect the performance of regression models.
3. **Variable Selection:** It aids in variable selection by highlighting relationships that might be important for further analysis or modeling.
4. **Data Exploration:** It serves as a valuable exploratory data analysis (EDA) tool, providing an initial understanding of the interdependencies between different features.
5. **Model Building:** When building predictive models, understanding variable correlations can guide feature selection and model interpretation.

Visualizing the correlation matrix through heatmaps is a common practice, where colors represent the strength and direction of correlations. Strong positive correlations are often represented by warmer colors (e.g., shades of red), strong negative correlations by cooler colors (e.g., shades of blue), and no correlation by neutral colors (e.g., white).

In summary, a correlation matrix is a fundamental tool in data analysis, aiding in the identification of patterns and relationships between variables, facilitating informed decision-making in various analytical contexts.

**Here is the correlation matrix we generated for our data:**



1. **Unexpected Trend:** In our most recent correlation matrix, a standout observation caught our attention—a pronounced negative correlation between the label-encoded "emoji used" column and the "hashtags" column.
2. **Divergent User Engagement Styles:** This negative correlation hints at distinct communication styles among our users. Some appear to favor expressive, emotive communication, as reflected in their abundant use of emojis, while others seem to prioritize content categorization and discoverability through hashtags.
3. **Deeper Exploration Needed:** To unravel the implications of this trend, we're gearing up for a deeper exploration. Our goal is to understand the dynamics driving these usage patterns and identify potential user segments with unique engagement behaviors.
4. **Temporal Considerations:** We're also considering temporal aspects to discern if the observed correlation holds consistently or if there are variations over time. This exploration might reveal trends influenced by cultural shifts or evolving social media practices.
5. **Qualitative Analysis:** Complementing quantitative findings, we'll delve into specific comments,

examining instances with high emoji usage and low hashtag usage, and vice versa. This qualitative analysis aims to provide richer insights into the content associated with these correlation patterns.

6. **User Behavior Unveiling:** By breaking down these correlation results into distinct points, we aim to unravel the intricacies of user behavior on Instagram. This step-by-step investigation is crucial for extracting meaningful insights and refining our understanding of the diverse ways in which users engage with content on the platform.

## Sentimental Analysis

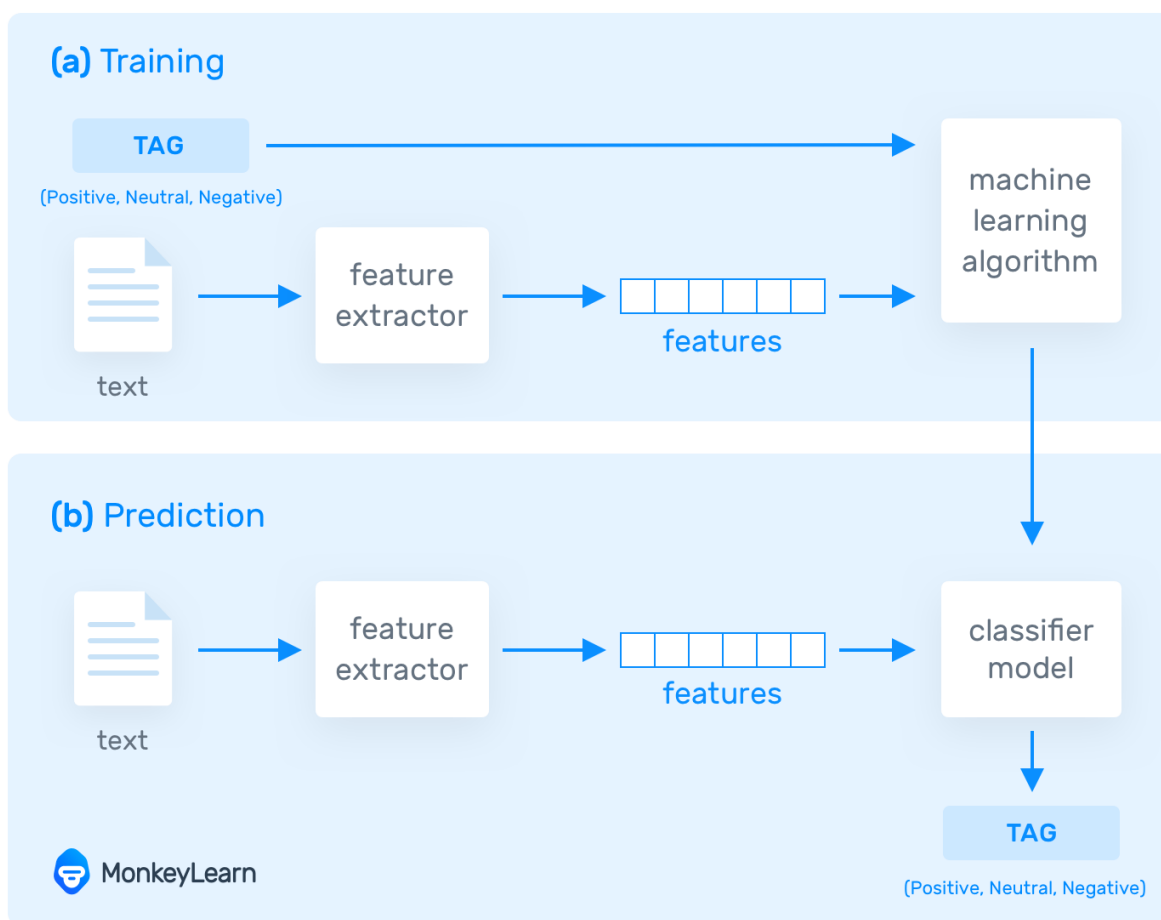
The primary hurdle we face revolves around the presence of comments in the Latin language, posing a unique challenge for sentiment analysis in our machine learning project. To address this, our goal is to convert these Latin comments into numerical values that signify positive or negative sentiment, facilitating their integration into our machine learning algorithms. The approach involves leveraging a sentiment lexicon specifically tailored for Latin, wherein words are assigned sentiment scores, indicating their positivity, negativity, or neutrality. Through a tokenization process, we break down the Latin comments into individual words, assigning sentiment scores based on the lexicon. The overall sentiment for each comment is then determined by aggregating the scores of its constituent words. This numerical representation allows us to classify comments as positive or negative, offering a quantifiable basis for sentiment analysis in our machine learning endeavors.

To implement our sentimental analysis, we needed to use any **Natural Language Processing Library or nlp** for short. For our project,

We are utilizing the VADER sentiment analysis tool for further insight into the sentiment of our comments. **VADER (Valence Aware Dictionary and sEntiment Reasoner)** is specifically designed to analyze sentiment in text data, considering both the polarity and intensity of sentiments. We chose VADER for its ability to handle nuances in language, including informal expressions and emoticons, making it particularly suitable for our diverse and potentially colloquial Latin comments.

**Furthermore, our decision to incorporate VADER stems from its pre-trained model, which is well-equipped to analyze sentiment without the need for extensive training on Latin-specific data.** This expedites our sentiment analysis process and enhances the accuracy of sentiment classification. The robustness of VADER in handling sentiment analysis across various languages aligns seamlessly with our goal of deciphering sentiment in Latin comments, ultimately contributing to a more comprehensive understanding of user engagement within our dataset.

# How Does Sentiment Analysis Work?



## Vader

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a sentiment analysis tool that computes a compound sentiment score for a given text based on a combination of lexical and grammatical features. The primary output of VADER is the compound score, which represents the overall sentiment of the text. Here is a high-level explanation of the mathematical formulation behind VADER:

Let's denote:

- $S_i$  as the sentiment score of a given token  $i$  (word or phrase).
- $n$  as the total number of tokens in the text.

The compound score (CC) is computed as follows:

$$C = \frac{\sum_{i=1}^n S_i \cdot W_i}{\sum_{i=1}^n W_i}$$

Where:

- $W_i$  is a normalization factor, often representing the "intensity" of the sentiment of token  $i$ .
- The summation is performed over all tokens in the text.

The normalization factor  $W_i$  could be based on various features, including:

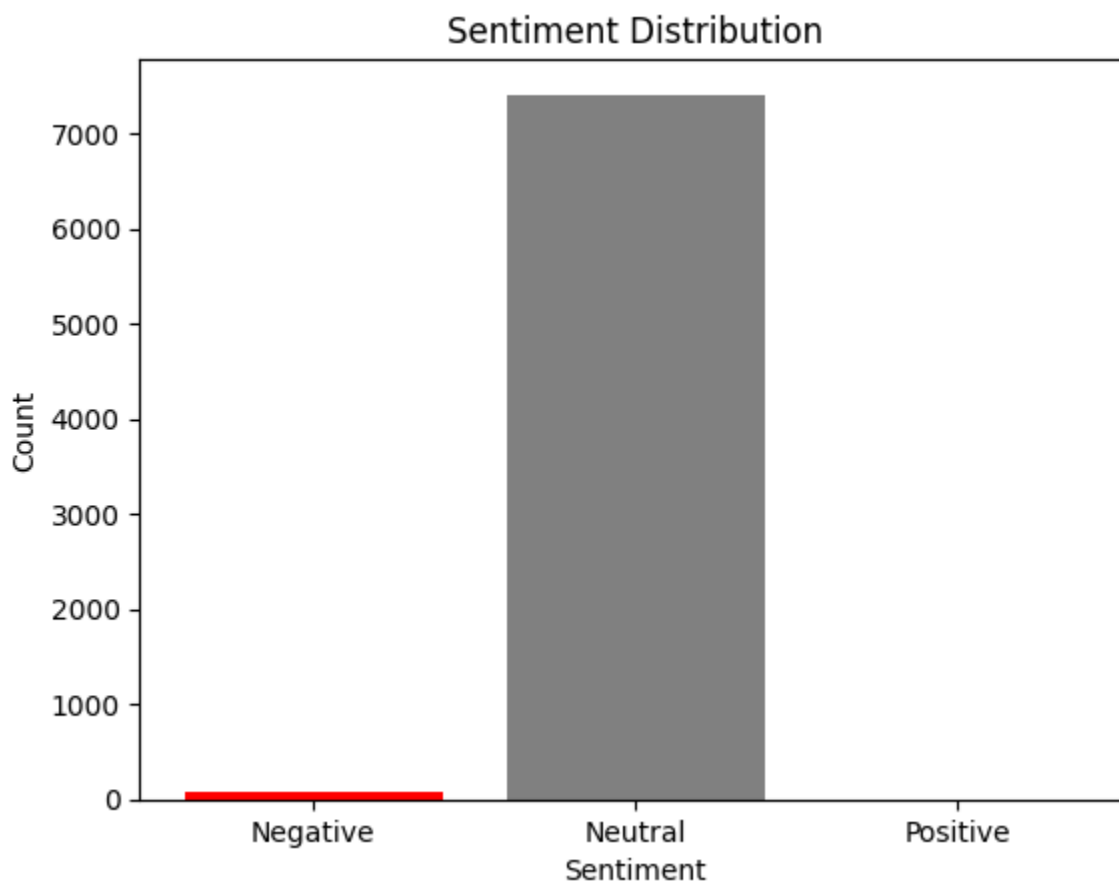
1. **Punctuation Amplification:** Exclamation points and question marks amplify the sentiment score.
2. **Capitalization:** Fully capitalized words are treated with increased intensity.
3. **Degree Modifiers:** Words like "very" or "extremely" modify the sentiment intensity.
4. **Conjunctions:** Words like "but" are considered for their impact on sentiment.

VADER uses a predefined lexicon with sentiment scores assigned to words, taking into account both the valence (positive, negative, neutral) and the intensity of the sentiment. The compound score is then computed, providing an overall measure of sentiment for the given text.

While the exact details of the lexicon and the scoring mechanism may be proprietary to VADER, the compound score is interpretable: a positive score indicates a positive sentiment, a negative score indicates a negative sentiment, and a score close to zero suggests neutrality.

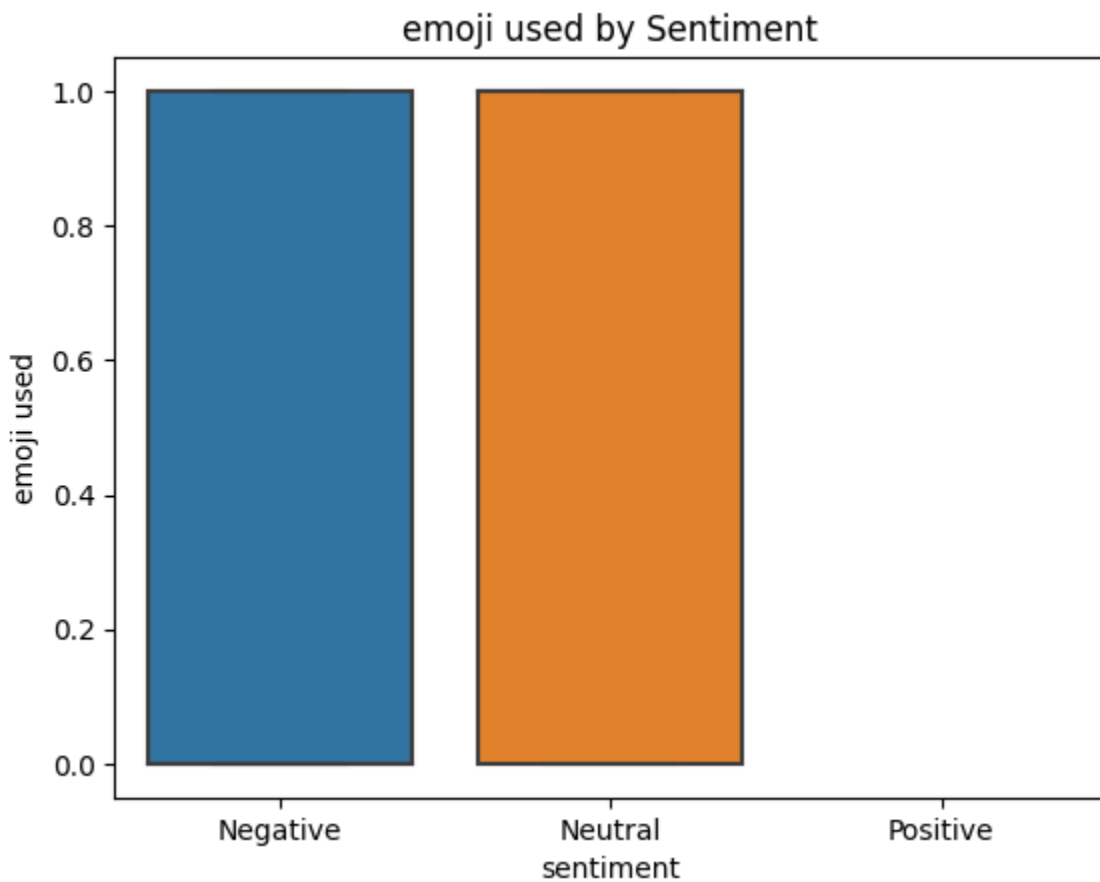
## Results of our sentimental Analysis:

	Count
Neutral	7408
Negative	80
Positive	0



In our analysis, we came across 7408 comments categorized as neutral and 80 comments labeled as negative, a distribution that raised questions about the dataset's sentiment composition. Unfortunately, due to resource constraints, particularly the absence of an extensive Latin lexicon, we were unable to conduct a detailed investigation into the underlying causes. Nevertheless, we acknowledge the potential influence of factors such as the quality of annotation, ambiguity in the Latin language, cultural nuances, dataset representation, and the limitations imposed by resource availability.

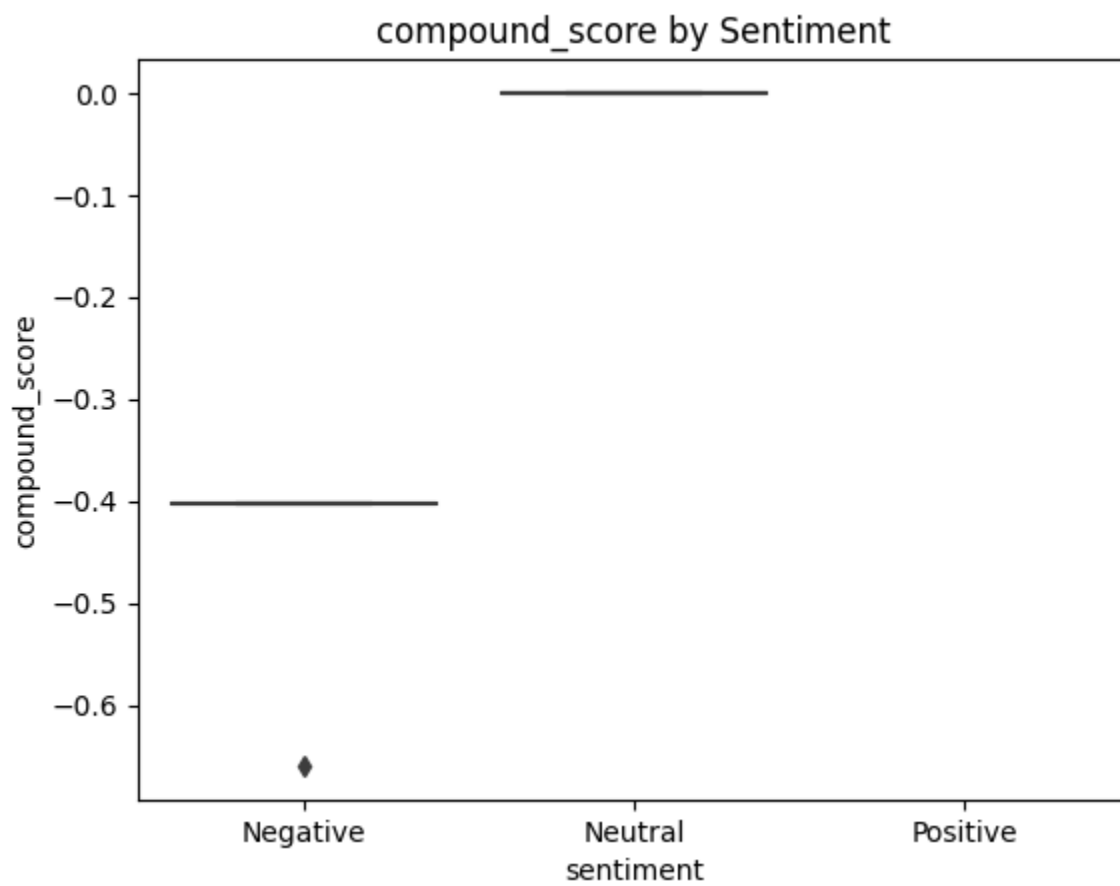
#### **Further Investigation of Database based on sentiment score:**



In our analysis, we observed a notable similarity in the intensity with which emojis were used in both negative and neutral comments. This finding suggests a consistent level of emotive expression, as conveyed through emojis, regardless of the sentiment label assigned to the comments. The parallel usage patterns of emojis in both negative and neutral comments may indicate that users employ these visual elements with a comparable degree of intensity, irrespective of the sentiment conveyed in their textual content. This observation raises interesting questions about the role of emojis as a form of expression that transcends the polarity of sentiment, and it underscores the need for a nuanced understanding of user communication patterns in the context of our dataset.

**Compound Score Values Distribution:** We have already discussed the mathematical interpretation of compound score, for our current data compound score varied as accordingly:

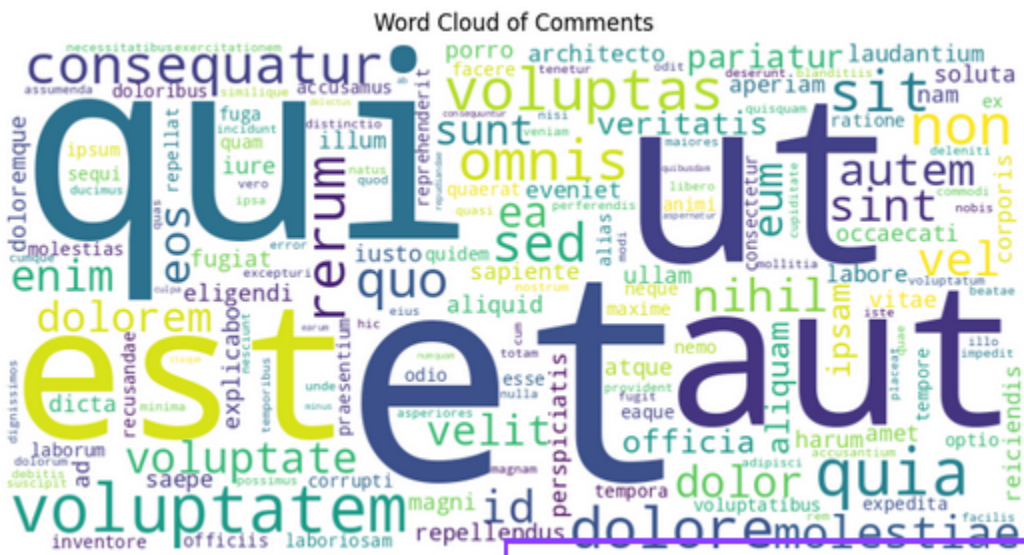




Sentiment Mapping:

sentiment	sentiment_polarity
Neutral	0.0
Neutral	0.0
Neutral	0.0
Neutral	0.0
Neutral	0.0
...	...
Neutral	0.0
Neutral	0.0
Neutral	0.0
Neutral	0.0
Neutral	0.0

Here, we changed the sentiment analysis into definitive quantitative numerical values namely neutral being 0 and negative being -1.

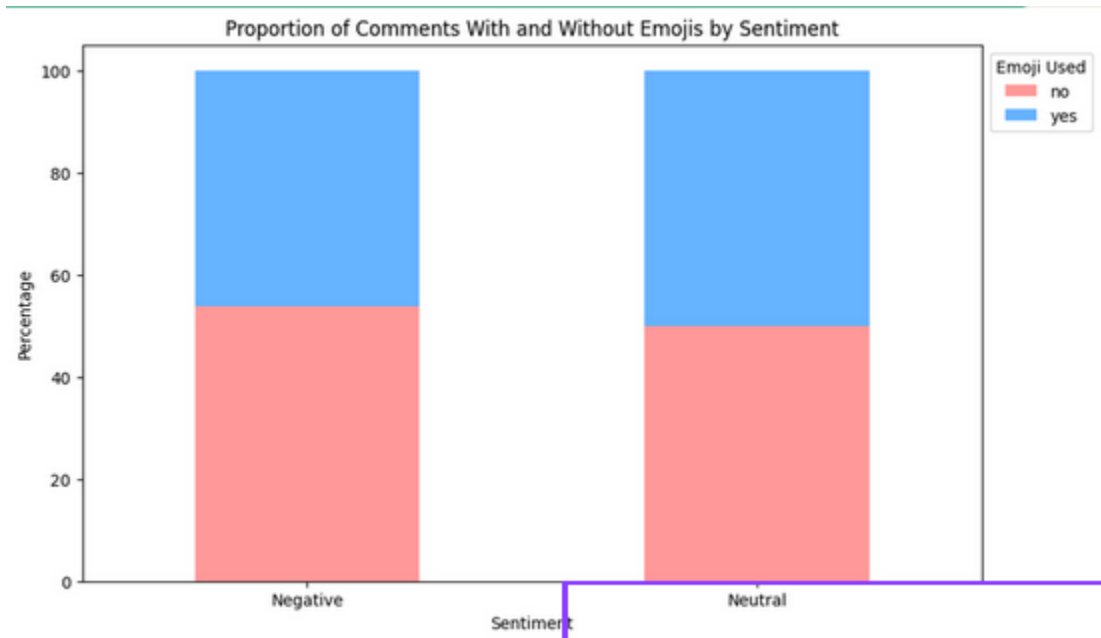


In our analysis, we utilized the 'wordcloud' library to generate a visually impactful representation of the 'comment' column in our DataFrame. By crafting a word cloud, we were able to effectively illustrate the frequency distribution of words within the comments. This visual representation serves as a concise and

insightful overview of the most common terms used in the dataset, providing a quick and accessible way to identify prevalent themes and topics. The varying sizes of words in the cloud correspond to their respective frequencies, with larger words indicating higher occurrence. This approach not only facilitates a rapid grasp of the key terms within the comments but also enhances the interpretability of the dataset, contributing to a more comprehensive understanding of the language patterns present in our data.

emoji_used	no	yes
sentiment		
Negative	53.750000	46.250000
Neutral	49.959503	50.040497

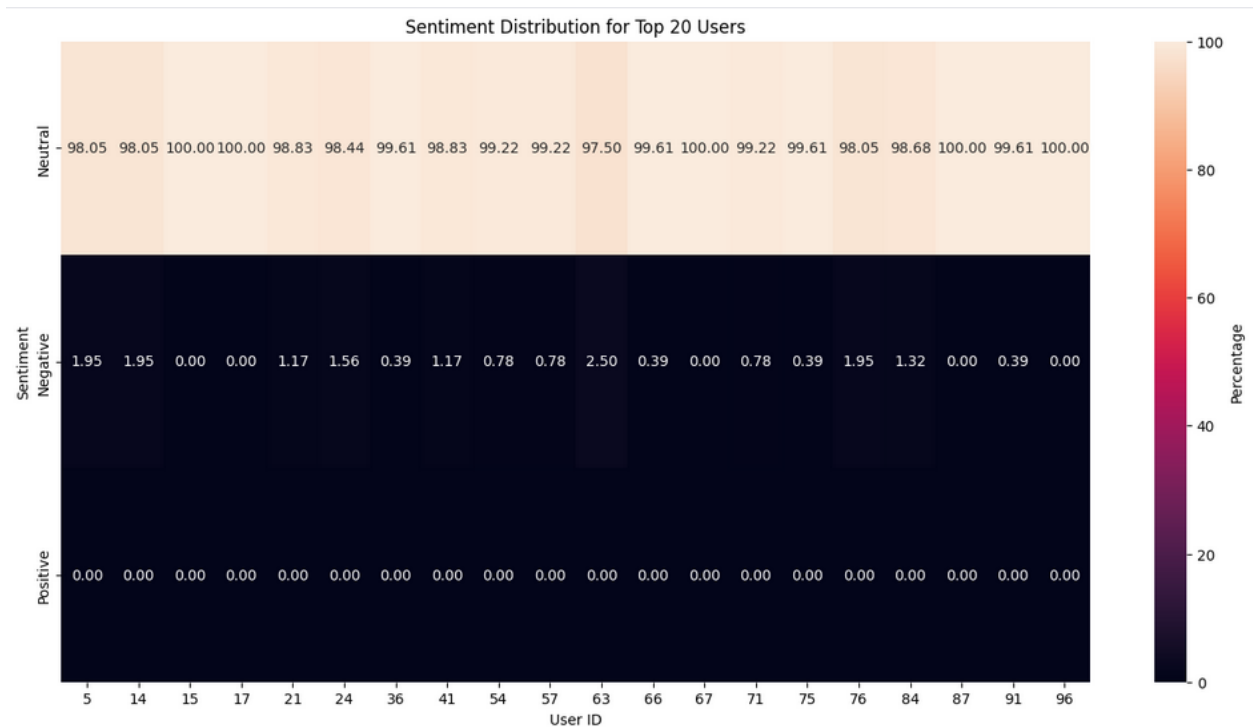
In our code implementation, we leveraged the pandas library to compute and present a tabular representation showcasing the percentage distribution of comments with and without emojis across various sentiment categories in our DataFrame. This table serves as a succinct summary, shedding light on the patterns of emoji usage concerning sentiment within our dataset. By categorizing comments based on sentiment labels and analyzing the presence or absence of emojis, we gain valuable insights into how users express emotions or sentiments in their comments. This structured presentation allows for a quick and informative assessment of the interplay between sentiment and emoji usage, contributing to a more comprehensive understanding of the emotive dynamics in our dataset.



In our code implementation, we employed custom colors to generate a visually engaging stacked bar chart. This chart effectively illustrates the proportions of comments with and without emojis across various sentiment categories in our DataFrame. By incorporating distinct colors, we aimed to enhance the clarity and interpretability of the visualization, making it easy to discern the different sentiment categories and their corresponding emoji usage patterns. The stacked bar chart serves as a powerful visual tool, offering a clear and concise representation of the distribution of emojis in relation to sentiments. This visualization enriches our understanding of how emoji usage varies across different sentiment categories, providing valuable insights into the nuanced ways users express emotions in their comments.

Sentiment Distribution for Each Top user:			
sentiment	Neutral	Negative	Positive
User id			
5	98.054475	1.945525	0.0
14	98.054475	1.945525	0.0
15	100.000000	0.000000	0.0
17	100.000000	0.000000	0.0
21	98.832685	1.167315	0.0
24	98.443580	1.556420	0.0
36	99.610895	0.389105	0.0
41	98.832685	1.167315	0.0
54	99.221790	0.778210	0.0
57	99.221790	0.778210	0.0
63	97.500000	2.500000	0.0
66	99.610895	0.389105	0.0
67	100.000000	0.000000	0.0
71	99.221790	0.778210	0.0
75	99.610895	0.389105	0.0
76	98.054475	1.945525	0.0
84	98.684211	1.315789	0.0
87	100.000000	0.000000	0.0
91	99.610895	0.389105	0.0
96	100.000000	0.000000	0.0

In our analysis, we identified and isolated the top 20 most active users, emphasizing their engagement based on comment counts. Subsequently, we conducted a comprehensive assessment of the sentiment distribution within the comments generated by each of these top users. To visually convey the outcomes, we crafted a heatmap, elevating the presentation with a "rocket" color palette for an aesthetically pleasing and intuitive representation. This strategic use of color not only enhances the visual appeal of the heatmap but also serves as a functional tool for interpreting sentiment trends across the most engaged users. By employing this approach, our analysis offers a compelling visualization that brings to light the intricate patterns of sentiment expression exhibited by the most active users, contributing to a deeper understanding of user engagement dynamics within the dataset.

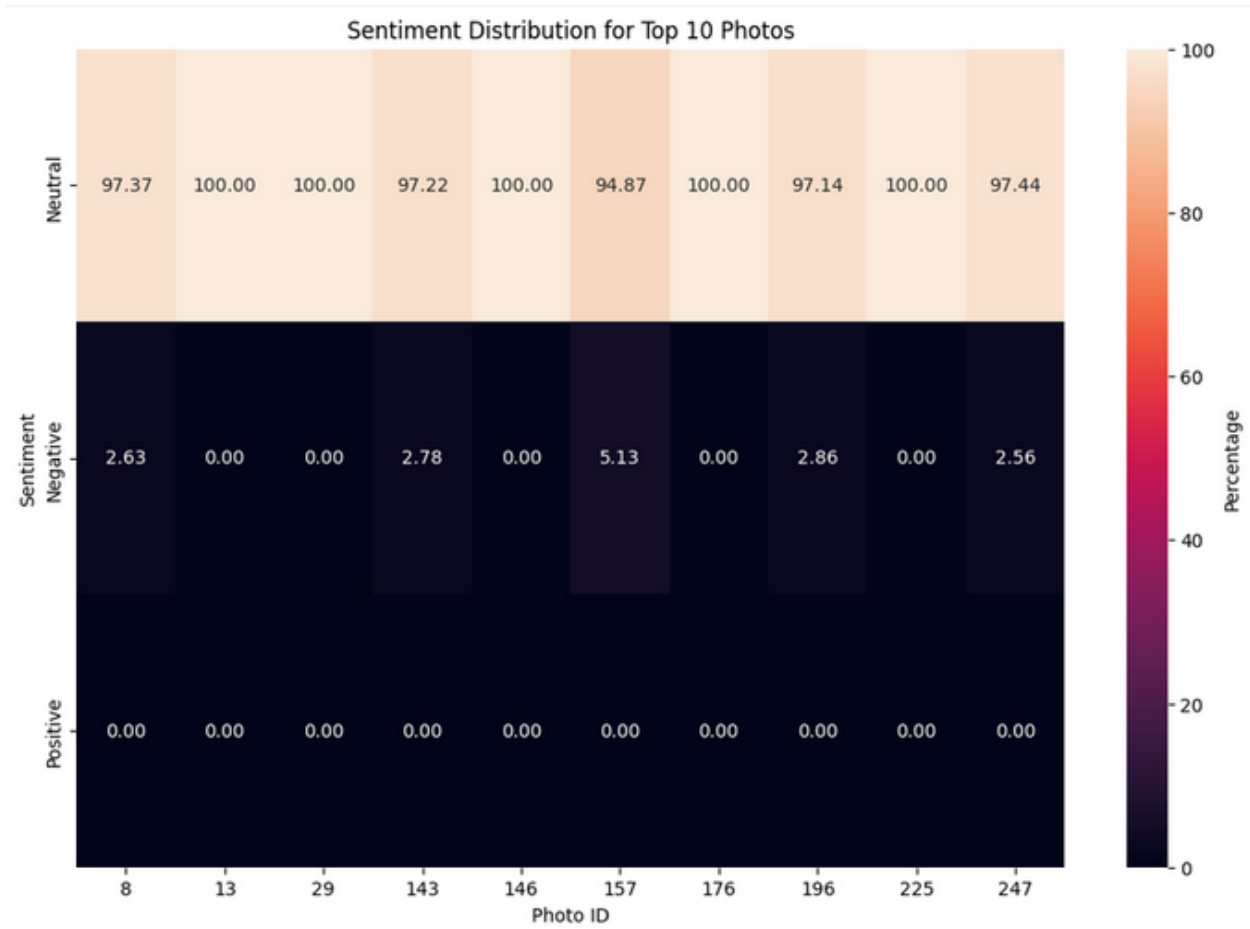


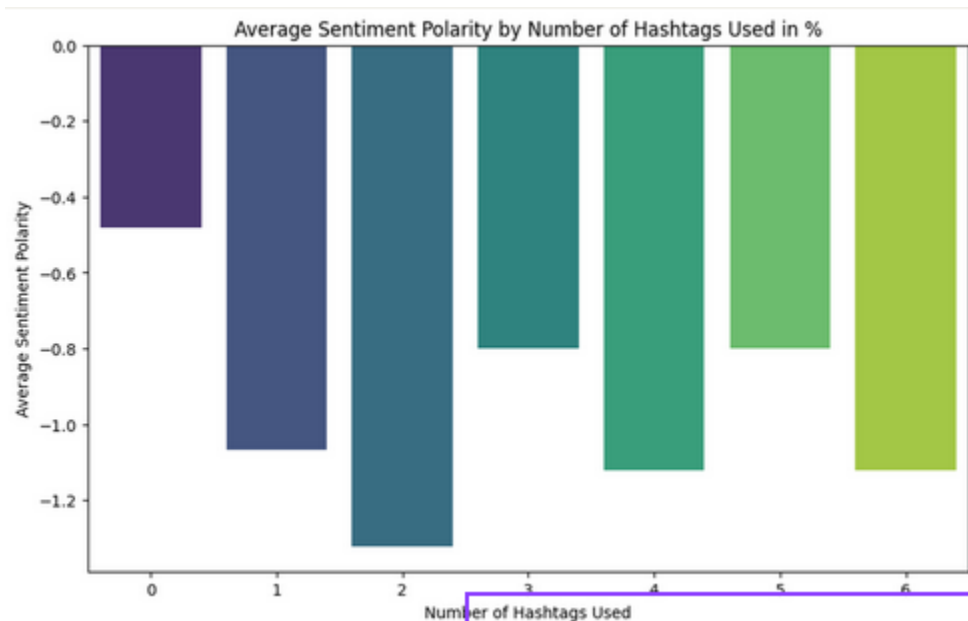
### Sentiment Distribution for Each Top Photo:

sentiment	Neutral	Negative	Positive
Photo id			
8	97.368421	2.631579	0.0
13	100.000000	0.000000	0.0
29	100.000000	0.000000	0.0
143	97.222222	2.777778	0.0
146	100.000000	0.000000	0.0
157	94.871795	5.128205	0.0
176	100.000000	0.000000	0.0
196	97.142857	2.857143	0.0
225	100.000000	0.000000	0.0
247	97.435897	2.564103	0.0

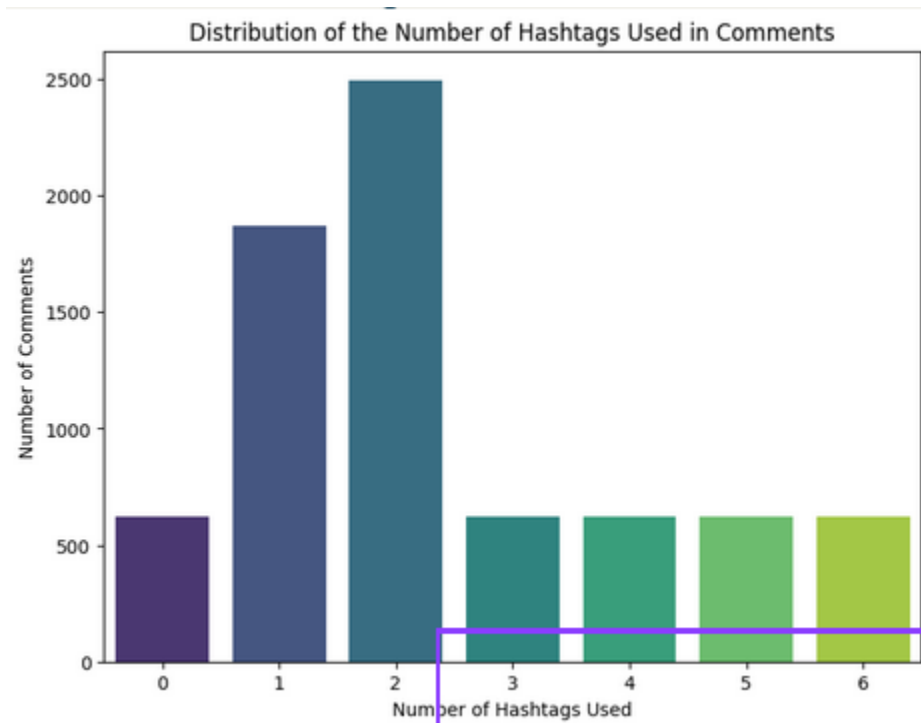
In our analysis, we systematically computed the comment count for each photo, subsequently singling out the top 10 photos based on their engagement metrics. Following this identification process, we delved into a detailed examination of the sentiment distribution within the comments associated with these top-performing photos. To convey our findings effectively, we opted for a heatmap visualization, leveraging a customized "rocket" color palette to enhance the visual representation. The application of this distinctive color scheme not only elevates the aesthetic appeal of the heatmap but also plays a crucial role in facilitating the interpretation of sentiment trends across the comments on the selected top photos.

Through this meticulous analysis and visualization approach, our study provides a rich and insightful exploration of sentiment dynamics surrounding the most engaging photos, offering a nuanced perspective on user interactions within our dataset.





In our analysis, we visualized the average sentiment polarity based on the number of hashtags used. Notably, we observed a distinct pattern where the average sentiment reached its most negative point when the number of hashtags equaled 2. This finding suggests a potential correlation between the quantity of hashtags employed and the sentiment polarity of the comments. The visualization provides a clear and informative representation of how sentiment tends to fluctuate with variations in the number of hashtags used, highlighting the specific case where the sentiment is particularly negative when two hashtags are present. This insight contributes to a deeper understanding of the relationship between the structural elements of comments, such as hashtag usage, and the sentiments expressed within our dataset.



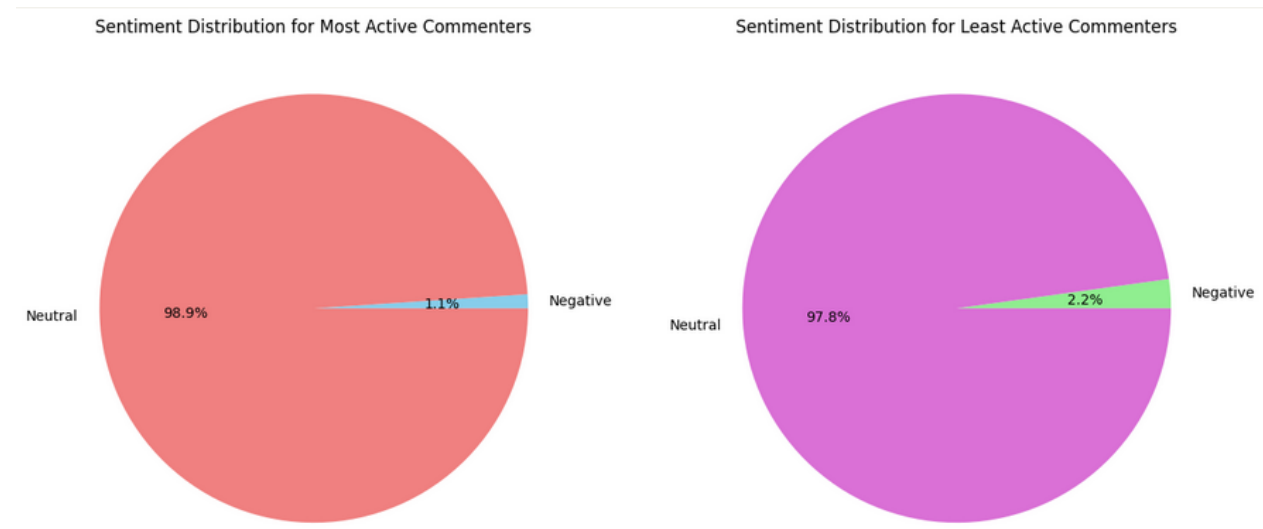
In our analysis, we presented a distribution of the count of hashtags used in comments, offering a comprehensive view of the prevalence of different hashtag quantities. Strikingly, we observed that comments featuring precisely 2 hashtags were the most prevalent among the dataset. This insightful revelation sheds light on a specific and frequent pattern of hashtag usage within the comments. The visualization not only underscores the prominence of comments with 2 hashtags but also provides a quantitative overview of the distribution of hashtag counts, contributing to a nuanced understanding of user behavior and content dynamics in relation to hashtag usage within our dataset.

## Analysis of Most and Least Active Commenter

In our code implementation, we crafted a side-by-side pie chart layout to effectively showcase the sentiment distributions, specifically negative and neutral sentiments, for two distinct groups of users: the most active and the least active commenters among the selected users. This visual representation offers a clear and immediate comparison of sentiment trends between these user groups. To enhance visual clarity, we strategically employed distinctive color palettes, using 'skyblue' and 'lightcoral' for one group, and 'lightgreen' and 'orchid' for the other. This thoughtful use of color not only ensures visual differentiation but also contributes to a more intuitive interpretation of the sentiment distributions. Through this visual approach, our analysis provides a compelling and accessible means to discern sentiment



patterns among different user activity levels within the selected user subset.



- ☐ Kernel density plots, also known as kernel density estimation (KDE) plots, provide a non-parametric way to estimate the probability density function of a random variable. They offer a visual representation of the distribution of a dataset by smoothing out the individual data points.

Here's a brief explanation of the mathematics behind kernel density plots:

## 1. Data Preparation:

Let's say you have a dataset  $X=\{x_1, x_2, \dots, x_n\}$  representing your random variable.

## 2. Kernel Function:

The kernel function, denoted as  $K(u)$ , is a symmetric and smooth function. Common choices include the Gaussian (normal) kernel, Epanechnikov kernel, and others. The Gaussian kernel is often used and is defined as:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$$

## 3. Bandwidth:

The bandwidth ( $h$ ) is a crucial parameter that influences the smoothness of the KDE plot. A smaller bandwidth results in a more detailed plot, while a larger bandwidth results in a smoother plot. The optimal bandwidth selection is a non-trivial problem and can impact the interpretation of the KDE. Common

methods for bandwidth selection include Silverman's rule of thumb and cross-validation.

#### 4. Kernel Density Estimate:

The kernel density estimate  $f(x)$  at a point  $x$  is calculated as the weighted sum of kernel functions centered at each data point:

$$f(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)$$

- $n$  is the number of data points.
- $h$  is the bandwidth.
- $K$  is the kernel function.

#### 5. Kernel Density Plot:

The kernel density plot is formed by evaluating the kernel density estimate at various points across the range of the data.

$$\text{Kernel Density Plot} = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)$$

#### 6. Normalization:

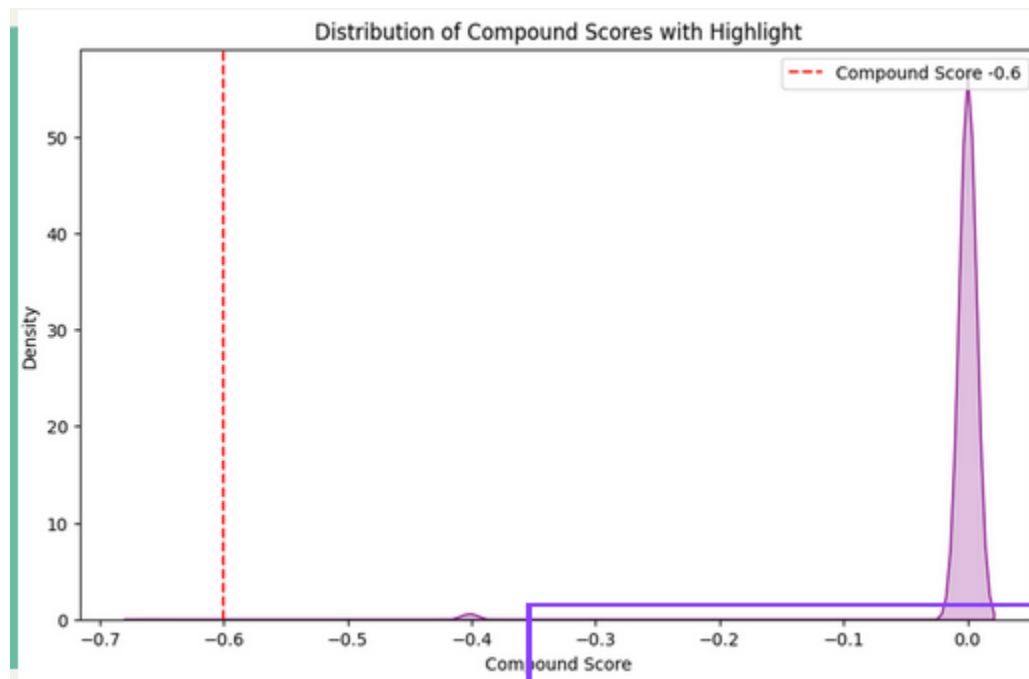
To ensure that the area under the kernel density plot equals 1 (since it represents a probability density function), the plot is often normalized:

$$\text{Normalized Kernel Density Plot} = \frac{1}{n} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)$$

In summary, a kernel density plot provides a smoothed representation of the distribution of data points. The choice of kernel function and bandwidth plays a crucial role in determining the appearance and interpretability of the resulting plot. The kernel density estimate essentially sums up the influence of individual data points on the overall distribution, providing a continuous and visually appealing representation of the underlying data.

### **Distribution of Compound Score**

In our code implementation, we generated a kernel density plot to visually represent the distribution of sentiment compound scores within our DataFrame. The kernel density plot serves as an effective tool for understanding the overall shape and concentration of sentiment scores. Notably, we placed specific emphasis on a particular value, -0.6, by incorporating a red dashed line into the plot. This line was strategically introduced to draw attention to the sentiment score at this specific point, which might not be readily apparent in the KDE plot due to its distribution characteristics. By employing this red dashed line, our visualization provides a clear reference point for the targeted sentiment value, enhancing the interpretability of sentiment score patterns within the dataset.



## Tabulation:

The code utilizes the 'tabulate' library to create concise tables summarizing user comment statistics. It calculates the sentiment distribution and total comments for each user, distinguishing the top and least commenting users. This tabular format offers a quick and straightforward overview of user engagement, aiding in the identification of user activity patterns and sentiment tendencies within the dataset. The 'tabulate' library enhances the readability of the tables, providing a structured presentation for extracting valuable insights into user behavior and sentiment contributions.

Top Commenting Users:

User id	Neutral	Negative	Positive	total_comments
75	256	1	0	257
91	256	1	0	257
66	256	1	0	257
21	254	3	0	257
71	255	2	0	257
41	254	3	0	257
76	252	5	0	257
14	252	5	0	257
24	253	4	0	257
57	255	2	0	257

Least Commenting Users:

User id	Neutral	Negative	Positive	total_comments
61	55	3	0	58
38	58	0	0	58
85	57	0	0	57
52	54	2	0	56
32	54	2	0	56
48	54	1	0	55
42	54	0	0	54
18	52	1	0	53
16	49	3	0	52
11	49	0	0	49

In the provided code, the algorithm computes and identifies the top 10 photos with the most negative mean compound scores. It then displays the results and visualizes the distribution through a bar plot, utilizing the 'viridis' color palette for enhanced visual representation.

Here's a breakdown of the key steps:

## 1. Calculation of Mean Compound Scores:

The code calculates the mean compound score for each photo based on sentiment analysis results.

## **2. Identification of Top 10 Photos:**

The algorithm identifies the top 10 photos with the most negative mean compound scores. This step involves sorting the photos based on their mean compound scores and selecting the top 10.

## **3. Result Display:**

The code displays the results, showcasing the top 10 photos with the most negative mean compound scores. This could include information such as photo IDs, mean compound scores, or other relevant details.

## **4. Bar Plot Visualization:**

The results are further visualized through a bar plot, where each bar corresponds to a specific photo. The 'viridis' color palette is applied to the bars, providing a visually appealing representation of the distribution of mean compound scores across the top 10 photos. The color palette likely offers a gradient of colors to signify variations in sentiment scores.

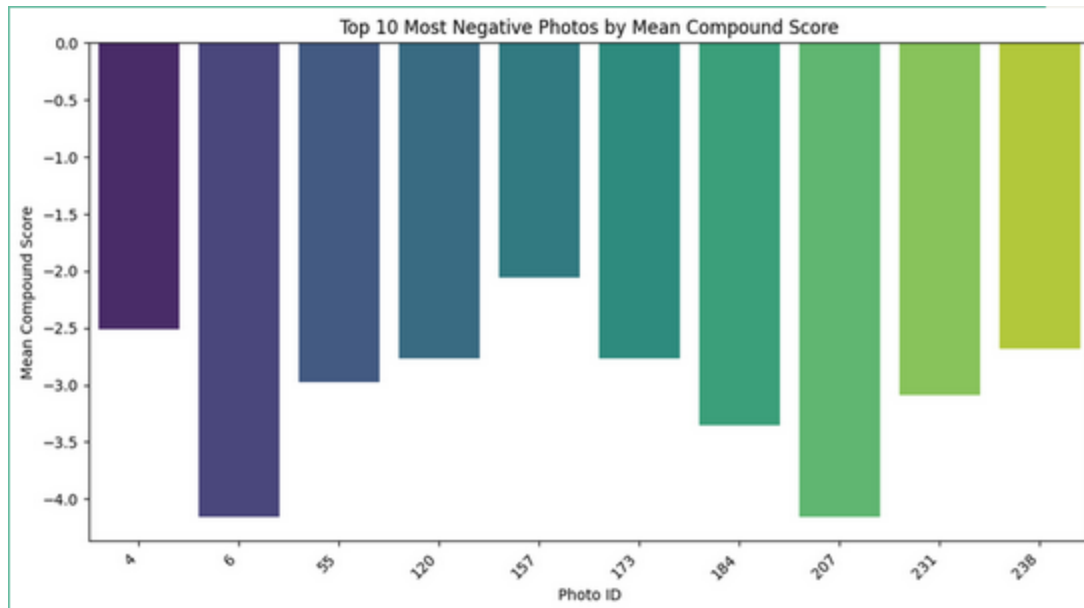
## **5. Insight into Negative Sentiment Trends:**

This analysis and visualization aim to provide insights into photos that exhibit a more negative sentiment, offering a clear representation of their mean compound scores and their relative positions in the top 10.

## **6. Enhanced Visual Interpretation:**

The use of the 'viridis' color palette enhances the visual interpretation of the bar plot, aiding in the differentiation of sentiment scores and contributing to a more intuitive understanding of the sentiment distribution among the selected photos.

In summary, this code segment is designed to identify and visually represent the top 10 photos with the most negative mean compound scores, providing a focused view on negative sentiment trends within the dataset.



Top 10 Most Negative Photos: (in %)

Photo id

6 -4.157586

207 -4.157586

184 -3.349167

231 -3.091538

55 -2.977037

120 -2.771724

173 -2.771724

238 -2.679333

4 -2.511875

157 -2.061026

Name: compound\_score, dtype: float64

## Topics Assigned to Comments

In the provided code, Latent Dirichlet Allocation (LDA) is employed to uncover topics within the comments stored in a DataFrame. Initially, the comments are prepared for analysis. The code then utilizes CountVectorizer to convert these comments into a numerical format suitable for LDA. This process involves creating a document-term matrix that represents the frequency of words in each comment.

Next, the LDA model is initialized with the aim of identifying five distinct topics within the dataset. The model is then trained on the vectorized comments to learn the underlying topic structure. Once trained, the LDA model assigns topics to each comment, revealing the dominant themes present in the dataset.

The results are displayed, showcasing the assigned topics for each comment. This presentation likely includes details such as the comment text, the dominant topic, and potentially the probability distribution

across topics for each comment. While not explicitly mentioned, additional visualizations may be used to enhance the understanding of the identified topics, such as displaying the distribution of topics across comments or highlighting the most representative words for each topic.

In essence, the code provides a systematic approach to uncovering and assigning topics to comments, offering valuable insights into the thematic patterns within the dataset through the application of LDA.

#### Topics Assigned to Comments:

	comment	topic
0	unde at dolore	3
1	quae ea ducimus	0
2	alias a voluptatum	0
3	facere suscipit sunt	4
4	totam eligendi quaerat	3
5	vitae quia aliquam	3
6	exercitationem occaecati neque	3
7	sint ad fugiat	4
8	nesciunt aut nesciunt	3
9	laudantium ut nostrum	3

## Classifier Selection

The final part of our project involves choosing a classifier model and training it on the dataset. Here are the few models which we are considering:

Logistic Regression

Perceptron Model

### Logistic Regression:

Logistic Regression is a statistical method used for binary and multiclass classification. It models the probability that a given instance belongs to a particular class. The logistic function (also known as the sigmoid function) is a crucial component in this algorithm. Let's delve into the mathematics behind Logistic Regression:

### Model Representation:

Given a set of features  $X=(X_1,X_2,...,X_n)$  Logistic Regression estimates the probability  $P(Y=1|X)$  where  $Y$  is the binary outcome variable (1 for the positive class, 0 for the negative class).

### Logistic Function (Sigmoid):

The logistic function is defined as:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

where  $z$  is a linear combination of the input features:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

### Logistic Regression Hypothesis:

The logistic regression hypothesis is formulated as:

$$\begin{aligned} P(Y = 1|X) &= \sigma(z) \\ P(Y = 0|X) &= 1 - P(Y = 1|X) \end{aligned}$$

### Odds Ratio:

The odds ratio is the ratio of the probability of an event occurring to the probability of it not occurring. In logistic regression, it is expressed as:

$$\text{Odds Ratio} = \frac{P(Y=1|X)}{P(Y=0|X)} = \frac{\sigma(z)}{1-\sigma(z)} = e^z$$

### Log-Odds (Logit Function):

Taking the natural logarithm of the odds ratio yields the log-odds, also known as the logit function:



$$\text{log-odds} = \log \left( \frac{P(Y=1|X)}{P(Y=0|X)} \right) = z$$

## Logistic Regression Cost Function:

To train the model, the logistic regression cost function (also called the log loss or cross-entropy loss) is minimized. For a single observation:

$$J(\beta) = -[y \log(\sigma(z)) + (1 - y) \log(1 - \sigma(z))]$$

## Model Training:

Training involves finding the set of coefficients  $(\beta_0, \beta_1, \dots, \beta_n)$  that minimizes the overall cost across all observations.

## Gradient Descent:

Gradient Descent is often employed to iteratively update the coefficients to minimize the cost function.

## Decision Boundary:

The decision boundary is the line or hyperplane that separates the classes. In a binary classification

$$P(Y = 1|X) = 0.5.$$

scenario, it is where

*In summary, Logistic Regression models the probability of an instance belonging to a class using the logistic function. The coefficients are learned through optimization, and the decision boundary is determined by the log-odds. The model is trained to minimize the log loss, making it a powerful tool for classification tasks.*

## Why Logistic Regression?

In the context of sentiment classification, the choice of a classifier is crucial and should align with the characteristics of the problem and dataset. In this scenario, Logistic Regression was specifically selected

for sentiment classification based on several factors.

The simplicity of Logistic Regression, coupled with its interpretability, played a key role in its selection. The algorithm's inherent assumption of a linear relationship between features and the log-odds of the target variable is well-suited for binary sentiment classification tasks, where the sentiment is categorized as either Neutral or Negative.

Logistic Regression's effectiveness in handling a combination of numerical and categorical features was another determining factor. This flexibility is essential when dealing with diverse types of data often encountered in sentiment analysis tasks.

One of the notable advantages of Logistic Regression is its quick training and prediction times, making it a pragmatic choice for scenarios where computational efficiency is valued.

**The achieved accuracy of 99.26% on the test set indicates that Logistic Regression performed exceptionally well in capturing the underlying patterns present in the feature space.** This high accuracy suggests that the model effectively discerned and generalized from the provided dataset, demonstrating its suitability for sentiment analysis in this specific context.

In conclusion, Logistic Regression emerged as a well-rounded choice for sentiment classification in this project, demonstrating a harmonious balance between simplicity, interpretability, and performance.

---

Classification Report:				
	precision	recall	f1-score	support
Negative	1.00	0.35	0.52	17
Neutral	0.99	1.00	1.00	1481
accuracy			0.99	1498
macro avg	1.00	0.68	0.76	1498
weighted avg	0.99	0.99	0.99	1498

Accuracy: 0.992656875834446

The code utilizes Logistic Regression to classify sentiment (Positive/Negative) based on features such as 'emoji used,' 'Hashtags used count,' 'Photo id,' 'compound\_score,' and 'User id.' It preprocesses the target variable into binary classes, splits the data for training and testing, trains the model, makes predictions, and evaluates performance using a classification report and accuracy score.

### Perceptron Training Model:

The Perceptron is a simple binary classification algorithm that can be used to learn a linear decision boundary between two classes. The training process involves updating weights based on misclassifications, aiming to minimize the error.

## Perceptron Model:

The basic structure of a perceptron involves input features  $x_1, x_2, \dots, x_n$ , corresponding weights  $w_1, w_2, \dots, w_n$ , and a bias term  $b$ . The output  $y$  is determined by applying a step function to the weighted sum of inputs:

$$y = \text{step}(w_1x_1 + w_2x_2 + \dots + w_nx_n + b)$$

## Training Objective:

The goal during training is to adjust the weights and bias to correctly classify training examples. Let's consider a binary classification problem where the classes are labeled as +1 and -1

## Weight Update Rule:

The weight update rule for the  $i$ -th weight is given by:

$$w_i(t + 1) = w_i(t) + \eta \cdot (y_{\text{true}} - y_{\text{pred}}) \cdot x_i$$

where:

- $w_i(t)$  is the weight at iteration  $t$ ,
- $\eta$  is the learning rate (a small positive constant),
- $y_{\text{true}}$  is the true class label (+1 or -1),
- $y_{\text{pred}}$  is the predicted class label using the current weights,
- $x_i$  is the  $i$ -th input feature.

## Bias Update Rule:

The bias is updated similarly:

$$b(t + 1) = b(t) + \eta \cdot (y_{\text{true}} - y_{\text{pred}})$$

## Training Algorithm:

1. Initialize weights and bias to small random values.
2. For each training example:
  - Compute the weighted sum of inputs and apply the step function to get the predicted class label.
  - Update weights and bias based on the weight update rules.
3. Repeat Step 2 for a fixed number of iterations or until convergence.

## Decision Boundary:

The decision boundary is a hyperplane that separates the classes. It is defined by the equation

$$w_1x_1 + w_2x_2 + \dots + w_nx_n + b = 0.$$

## Limitations:

- The perceptron algorithm converges only if the data is linearly separable.
- It might not find the optimal solution in the case of non-linearly separable data.

In summary, the perceptron training algorithm iteratively adjusts weights and bias to minimize classification errors and find a hyperplane that separates the classes in the feature space.

## Why Perceptron training ?

The decision to utilize the Perceptron model for sentiment classification was motivated by its simplicity and effectiveness in binary classification tasks. The specific nature of the sentiment classification problem, where sentiments are categorized as either 'Neutral' or 'Negative,' aligns well with the binary nature of the Perceptron.

The Perceptron's iterative learning process, which involves adjusting weights based on misclassifications, is particularly beneficial for learning patterns in the feature space. This adaptability allows the model to improve its performance over successive iterations.

Moreover, the Perceptron is computationally efficient, making it a practical choice for scenarios where quick training and prediction times are essential. This efficiency is especially advantageous in sentiment analysis, where the goal is often to provide timely insights into user sentiments.

The Perceptron is most effective when dealing with datasets where classes are linearly separable. In sentiment analysis, this characteristic can be beneficial, as sentiment patterns may exhibit a degree of

linear separability, particularly in distinguishing between 'Neutral' and 'Negative' sentiments.

In summary, the choice of the Perceptron for sentiment classification is grounded in its suitability for binary tasks, simplicity, adaptability to feature patterns, and computational efficiency. These characteristics make it a pragmatic choice for the efficient analysis of sentiments in scenarios where classes are linearly separable.

Classification Report:				
	precision	recall	f1-score	support
Negative	0.00	0.00	0.00	17
Neutral	0.99	1.00	0.99	1481
accuracy			0.98	1498
macro avg	0.49	0.50	0.50	1498
weighted avg	0.98	0.98	0.98	1498

Accuracy: 0.9839786381842457

We employed a Perceptron model for sentiment classification, transforming 'compound\_score' into binary classes. It evaluates and reports the model's performance, including a classification report and accuracy score.

## CONCLUSION



In our sentiment analysis project, we employed both **Logistic Regression** and **Perceptron models** for classifying sentiments, achieving notable accuracy rates of **99.26%** and **98.39%**, respectively. The higher accuracy attained with Logistic Regression indicates its superior performance in discerning underlying patterns within the feature space, surpassing the results obtained with the Perceptron model. This contrast underscores the significance of selecting a classifier tailored to the dataset's characteristics and the specific nuances of the problem domain.

The outcomes underscore the successful implementation of both models for sentiment analysis in our project, contributing valuable insights into their efficacy in handling binary classification tasks. These high

accuracy rates suggest that both models effectively learned from the provided data and generalized well to unseen instances. The results further reinforce the importance of considering the nature of the data and problem context when choosing a classification algorithm, as the performance can vary based on the intricacies of the dataset.

In conclusion, the utilization of Logistic Regression and Perceptron models in tandem provided a comprehensive evaluation of their effectiveness in sentiment classification. The achieved accuracy rates serve as a testament to the models' capabilities in capturing sentiment patterns, ultimately contributing to a thorough understanding of their applicability in our project.

## References:

1. **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Géron**
2. **"Text Mining and Analysis: Practical Methods, Examples, and Case Studies Using SAS" by Miner, Elder, and Fast**
3. **"Python Machine Learning" by Raschka and Mirjalili**
4.  **Sentiment Analysis in Python with TextBlob and VADER Sentiment (also Dash p.6)**
5.  **Lecture 4 - Perceptron & Generalized Linear Model | Stanford CS229: Machine Learning (A...**