Analyzing Sentiment Dynamics and Engagement Patterns

(Uncovering Collusion in Arbitrum Governance Forum with a Focus on Tally Proposal Posts)

Short Description of Main Task: The main objective of this task is to gain a deeper understanding of the key group of individuals actively involved in submitting proposals to the Arbitrum governance forum. The hypothesis suggests that there might be a certain group of individuals with shared interests who consistently support each other's proposals, potentially leading to a bias in decision-making and centralization of power. By conducting a thorough analysis of user interactions, sentiment dynamics, and engagement patterns, the aim is to uncover any signs of collusion and assess its impact on governance processes.

Subtask Description - Tally Proposal Post on Forum - Sentiment Analysis & Statistical Analysis:

In this subtask, the focus was on conducting sentiment analysis and statistical analysis on Tally proposal posts within the Arbitrum governance forum. Tally proposals play a crucial role in governance decisions within the community, making it essential to understand the sentiment dynamics surrounding these proposals. The objective was to identify prevailing sentiment trends, explore user engagement patterns, and assess the level of community support for Tally proposals. By analyzing sentiment in comments, examining trends over time, and correlating sentiment with engagement metrics, this subtask aimed to provide valuable insights into community sentiment and behavior towards Tally proposals.

Table Of Content:

Executive Summary:

In this analysis, we aimed to gain insights into the sentiment and engagement patterns surrounding Tally proposal posts on the forum. We conducted sentiment analysis on comments, investigated trends over time, examined sentiment differences across proposal categories, and explored user interactions.

Key findings include:

- The analysis revealed a predominantly positive sentiment in comments on Tally proposal posts, with positive sentiment significantly outweighing negative and neutral sentiments. This indicates a generally favorable view of Tally proposals among forum users.

- Sentiment towards Tally proposals has shown to be steady over time, with a consistent level of engagement from the community. This steadiness is crucial for maintaining a vibrant discussion environment around Tally proposals.

- Sentiment analysis demonstrated no significant difference in sentiment between Tally proposals and other proposal types on the forum. This finding suggests a uniform sentiment distribution across various discussion topics.

- Users across different trust levels contribute to discussions on Tally proposals equally, indicating a broad and inclusive participation base. This wide-ranging engagement is vital for the diversity of opinions and discussions.

- A clear correlation exists between a proposal's visibility (measured in views and likes) and the level of engagement (measured in comments) it receives. Proposals with higher visibility tend to engage more users, leading to a higher number of comments.

- The analysis did not find a strong correlation between the sentiment expressed in comments and the engagement metrics (views and likes). This suggests that while sentiment may influence some aspects of user engagement, other factors also play significant roles.

Introduction:

The analysis focuses on understanding sentiment and engagement dynamics surrounding Tally proposal posts on the forum. Tally proposals play a crucial role in governance and decision-making within the community. By analyzing sentiment in comments and exploring user interactions, we aim to gain insights into community sentiment, engagement trends, and potential areas for improvement in the governance process.

Methodology:

1. **Data Collection:** Utilized the Discourse API and SQL queries to extract forum data, including user profiles, topics, and posts. Tally proposal posts were identified based on specific topic IDs.

2. **Data Preprocessing:** Cleaned and prepared the data by merging relevant datasets, converting data types, and handling missing values.

3. **Analysis Approach:** Sentiment analysis was performed using the VADER sentiment analysis tool. Various visualization techniques, including bar charts, line charts, Sankey diagrams, and pie charts, were employed to explore and visualize the data.

Analysis Findings:

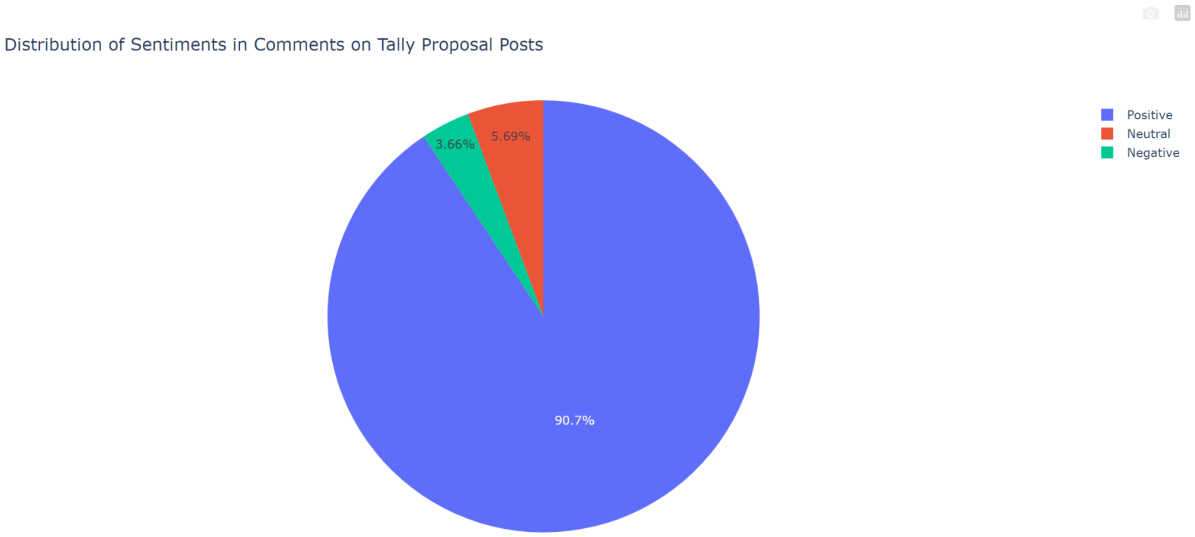
1. Dominant Sentiment in Comments on Tally Proposal Posts:

Objective: Determine the dominant sentiment (positive, negative, neutral) in comments.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    if scores['compound'] >= 0.05:
        return 'Positive'
    elif scores['compound'] <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Count the occurrences of each sentiment
sentiment_counts = tally_posts['Sentiment'].value_counts().reset_index()
sentiment_counts.columns = ['Sentiment', 'Count']

# Create an interactive pie chart using Plotly
fig = px.pie(sentiment_counts, values='Count', names='Sentiment',
             title='Distribution of Sentiments in Comments on Tally Proposal Posts')
fig.show()
```

Visualizations:-



Source:- Visualization Link

Visualization Explanation: A pie chart was utilized to display the distribution of sentiments in comments.

Insights: The visualization revealed a predominant positive sentiment, accounting for 90.7% of the comments, showcasing a highly favorable reception towards Tally proposals within the community. Neutral comments constituted 5.69%, indicating some level of ambivalence or reserved judgment, while negative sentiment was the least, at 3.66%, reflecting minimal opposition or dissatisfaction among the forum participants. This overwhelming positivity highlights the strong support and optimism within the community for the initiatives discussed in Tally proposals.

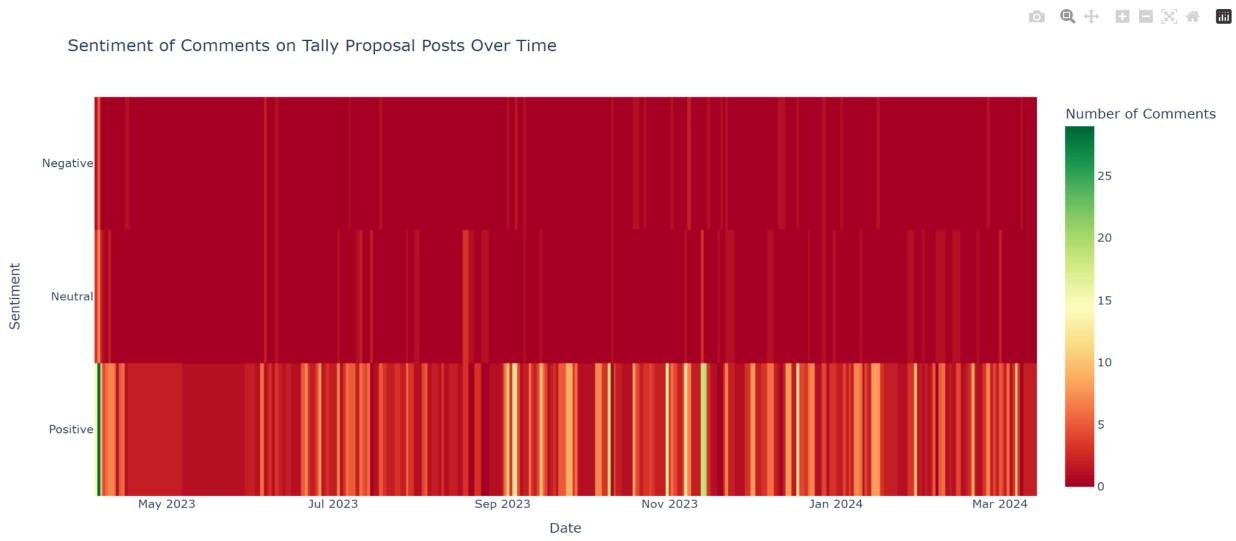
2. Trends in Sentiment Towards Tally Proposals Over Time:

Objective: Identify trends in sentiment (positive, negative) towards Tally proposals over time.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    sentiment = sid.polarity_scores(text)
    if sentiment['compound'] > 0.05:
        return 'Positive'
    elif sentiment['compound'] < -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Convert 'Post Created At' to datetime
tally_posts['Post Created At'] = pd.to_datetime(tally_posts['Post Created At'])

# Extract date and time components
tally_posts['Date'] = tally_posts['Post Created At'].dt.date

# Group by date and sentiment, and count the number of comments
sentiment_counts = tally_posts.groupby(['Date', 'Sentiment']).size().unstack(fill_value=0)

# Create a heatmap
fig = px.imshow(sentiment_counts.T,
                labels=dict(x='Date', y='Sentiment', color='Number of Comments'),
                x=sentiment_counts.index,
                y=sentiment_counts.columns,
                color_continuous_scale='RdYlGn',
                title='Sentiment of Comments on Tally Proposal Posts Over Time')

# Update layout
fig.update_layout(xaxis_title='Date', yaxis_title='Sentiment', coloraxis_colorbar=dict(title='Number of Comments'))

# Show the plot
fig.show()
```

Visualization:-



Source:- [Visualization Link]

Visualization Explanation: A heatmap was employed to visualize daily/weekly averages of sentiment scores.

Insights: The visualization effectively captured fluctuations, indicating a high volume of comments with predominantly positive sentiment in April 2023, which notably tapered off by May 2023. However, from September 2023 onwards, a consistent engagement level was observed, with an average of approximately 10 comments per day.

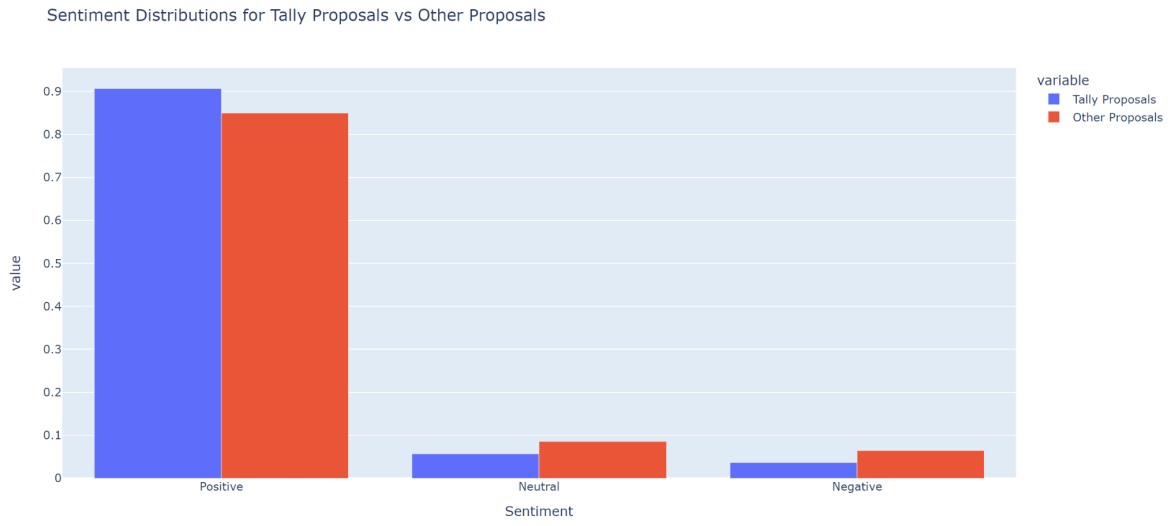
3. Sentiment Towards Tally Proposals vs. Other Proposals:

Objective: Determine if sentiment towards Tally proposals differs significantly from other proposals.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    if scores['compound'] >= 0.05:
        return 'Positive'
    elif scores['compound'] <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)
other_posts['Sentiment'] = other_posts['Post Description'].astype(str).apply(get_sentiment)

# Calculate sentiment distributions
tally_sentiment_counts = tally_posts['Sentiment'].value_counts(normalize=True)
other_sentiment_counts = other_posts['Sentiment'].value_counts(normalize=True)

# Perform statistical test (e.g., Chi-square test) to determine if there's a significant difference
chi2_stat, p_val, _, _ = stats.chi2_contingency([tally_sentiment_counts, other_sentiment_counts])

# Print results
print("Chi-square statistic:", chi2_stat)
print("P-value:", p_val)

# Plot sentiment distributions
fig = px.bar(pd.concat([tally_sentiment_counts, other_sentiment_counts], axis=1, keys=['Tally Proposals', 'Other Proposals']),
             barmode='group', title='Sentiment Distributions for Tally Proposals vs Other Proposals')
fig.show()
```

Visualization:-



Source:- [Visualization Link]

Visualization Explanation: A bar chart was used to visualize the Sentiment Towards Tally Proposals vs. Other Proposals

Insights: The comparative analysis through a bar chart demonstrated that the sentiment towards Tally proposals does not significantly differ from that of other proposals on the forum. This indicates a uniform perception or reception by the forum users towards various topics, suggesting that the content of proposals, whether Tally or otherwise, equally engages or elicits responses from the community.

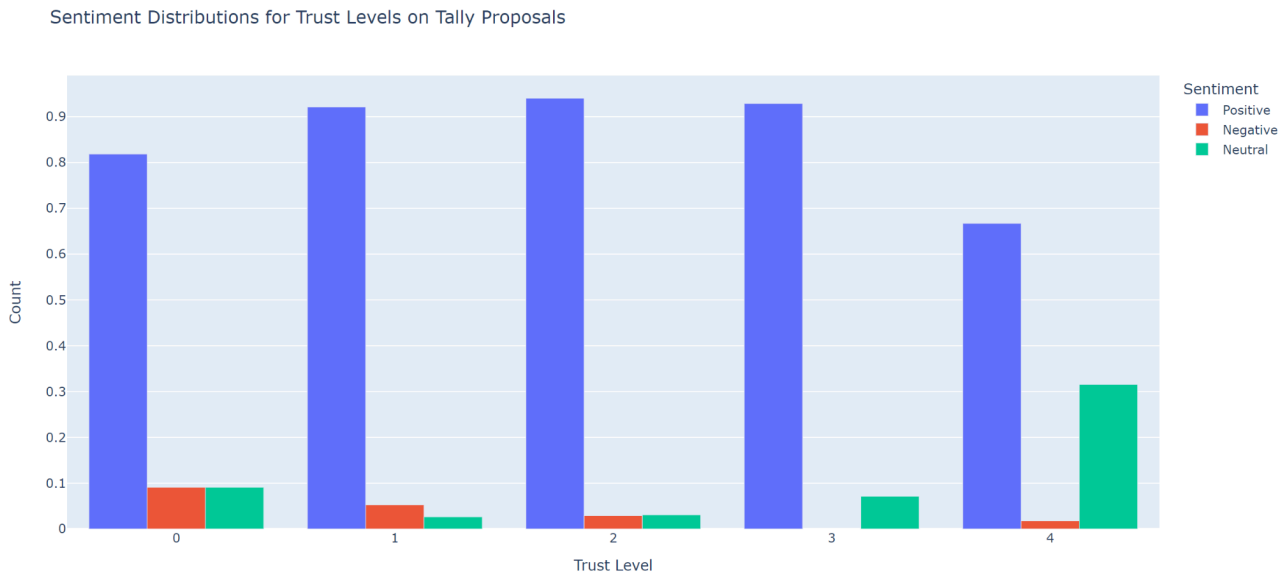
4. Sentiment of High "Trust Level" Users vs. Regular Users:

Objective: Compare sentiment expressions between high "Trust Level" users and regular users for Tally proposals.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    if scores['compound'] >= 0.05:
        return 'Positive'
    elif scores['compound'] <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Calculate sentiment distributions for high trust level users and regular users
trust_level_sentiment_counts = tally_posts.groupby('Trust Level')['Sentiment'].value_counts(normalize=True).reset_index(name='Count')

# Plot sentiment distributions
fig = px.bar(trust_level_sentiment_counts, x='Trust Level', y='Count', color='Sentiment',
             barmode='group', title='Sentiment Distributions for Trust Levels on Tally Proposals')
fig.show()
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A bar chart was utilized to compare sentiment distribution between user groups.

Insights: The analysis, depicted through a bar chart, reveals that users with Trust Levels 1, 2, and 3 are the most active in expressing sentiments towards Tally proposals, while those at Trust Level 4 show minimal sentiment expression. This suggests that engagement and sentiment expression on Tally proposals are not necessarily tied to a higher trust level.

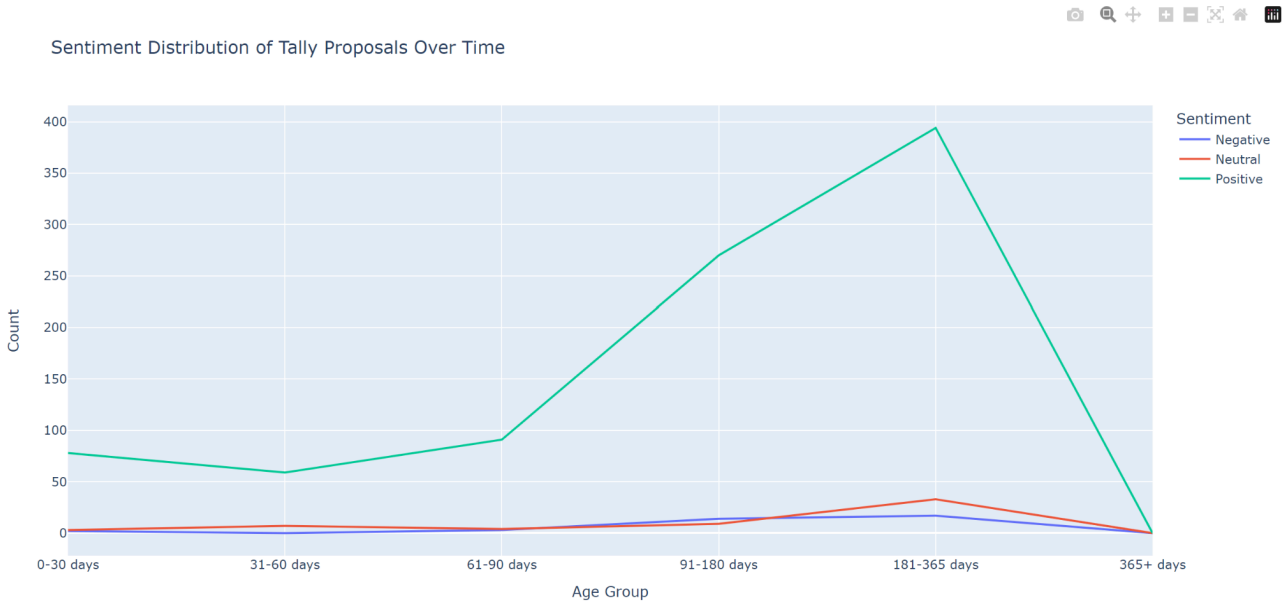
5. Change in Sentiment Over Time for Tally Proposals:

Objective: Examine if sentiment towards Tally proposals changes as they age on the forum.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    if scores['compound'] >= 0.05:
        return 'Positive'
    elif scores['compound'] <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Categorize proposals into different age groups (e.g., newly posted vs. older)
# You can adjust the age groups as per your requirement
tally_posts['Age Group'] = pd.cut((pd.Timestamp.now() - tally_posts['Post Created At']).dt.days, bins=[0, 30, 60, 90, 180, 365, float('inf')],
                                  labels=['0-30 days', '31-60 days', '61-90 days', '91-180 days', '181-365 days', '365+ days'])

# Calculate sentiment distributions for each age group
sentiment_distribution = tally_posts.groupby(['Age Group', 'Sentiment']).size().reset_index(name='Count')

# Plot sentiment distribution over time using line plot
fig = px.line(sentiment_distribution, x='Age Group', y='Count', color='Sentiment',
              title='Sentiment Distribution of Tally Proposals Over Time')
fig.show()
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A line chart was used to compare sentiment count between newly posted and older proposals.

Insights: The analysis highlights a significant trend where Tally proposals aged between 181 to 365 days exhibit the highest sentiment engagement, with approximately 394 sentiment counts, as shown in the line chart. In contrast, newly added proposals (0-30 days) witness the least sentiment engagement, with around 80 counts.

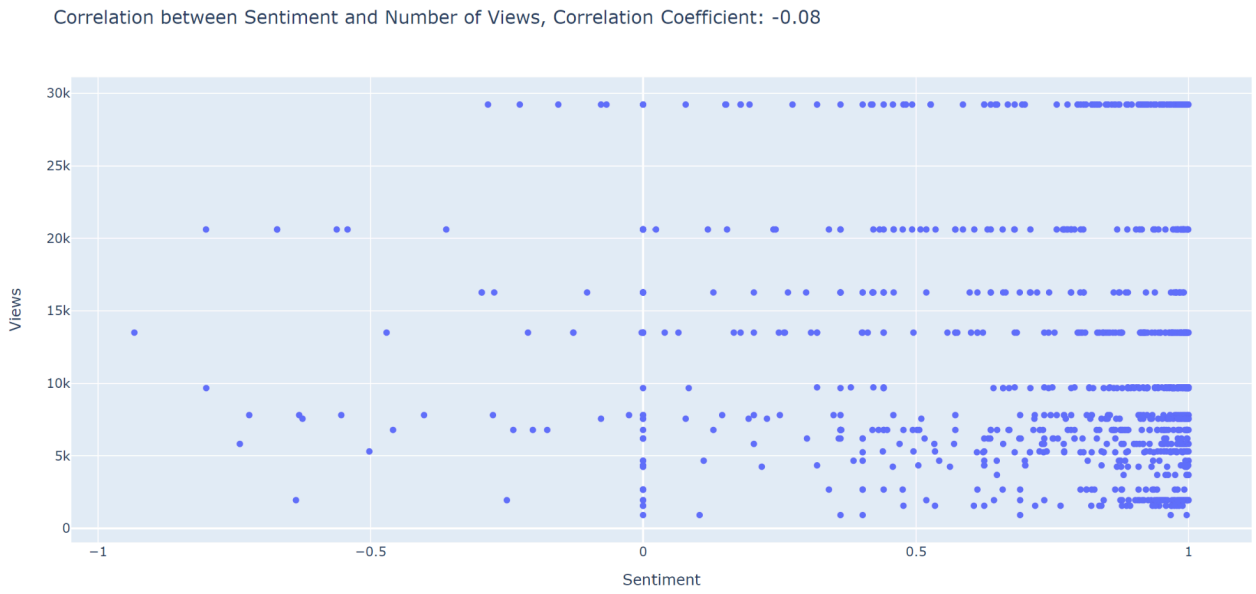
6. Correlation Between Sentiment and Number of Views:

Objective: Determine if there is a correlation between sentiment expressed in comments and the number of views a Tally proposal receives.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    return scores['compound']

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Calculate the correlation between sentiment and number of views
correlation = tally_posts[['Sentiment', 'Views']].corr().iloc[0, 1]

# Plot scatter plot with sentiment vs. number of views
fig = px.scatter(tally_posts, x='Sentiment', y='Views',
                 title=f'Correlation between Sentiment and Number of Views,\nCorrelation Coefficient: {correlation:.2f}')
fig.show()
```

Visualization:-



Source:- Visualization link

Visualization Explanation: A scatter plot was employed to visualize the relationship between sentiment scores and views.

Insights: The scatter plot visualization reveals a weak negative correlation (-0.08) between sentiment scores and the number of views received by Tally proposals. This indicates that as sentiment becomes more negative, there is a slight decrease in the number of views, and vice versa. While the correlation is weak, it suggests that sentiment expressed in comments may influence viewer engagement to some extent, albeit mildly.

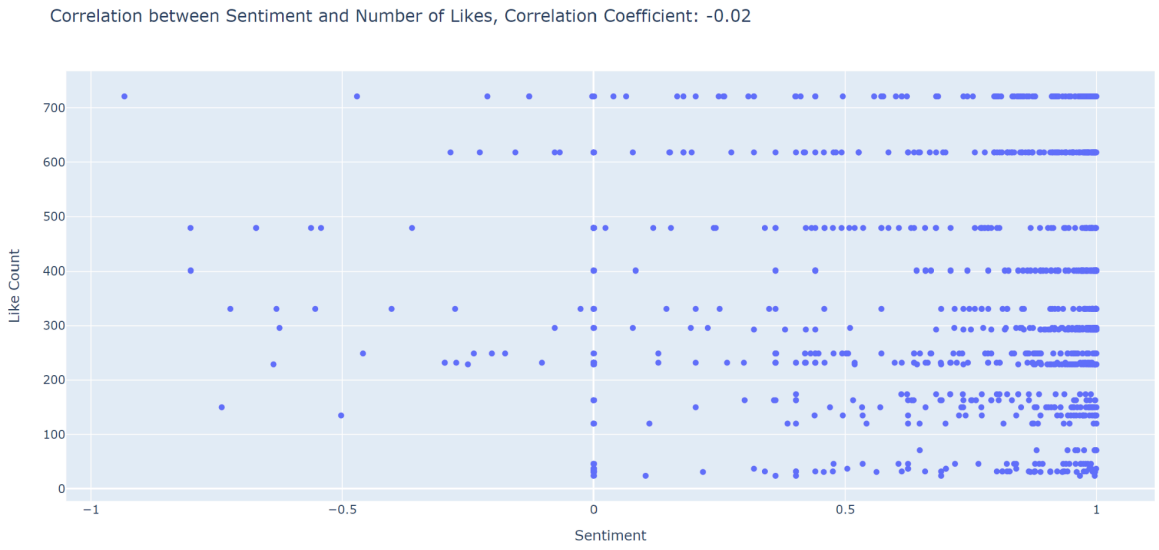
7. Correlation Between Sentiment and Number of Likes:

Objective: Explore the correlation between sentiment expressed in comments and the number of likes a Tally proposal receives.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    return scores['compound']

# Apply sentiment analysis to comments
tally_posts['Sentiment'] = tally_posts['Post Description'].astype(str).apply(get_sentiment)

# Calculate the correlation between sentiment and number of likes
correlation = tally_posts[['Sentiment', 'Like Count']].corr().iloc[0, 1]

# Plot scatter plot with sentiment vs. number of likes
fig = px.scatter(tally_posts, x='Sentiment', y='Like Count',
                 title=f'Correlation between Sentiment and Number of Likes,\nCorrelation Coefficient: {correlation:.2f}')
fig.show()
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A scatter plot was utilized to visualize the relationship between sentiment scores and likes.

Insights: Through the scatter plot, a negligible correlation coefficient of -0.02 is observed between sentiment scores and the number of likes received by Tally proposals. It can be observed that more users engage and like a positive sentiment expressed by the members.

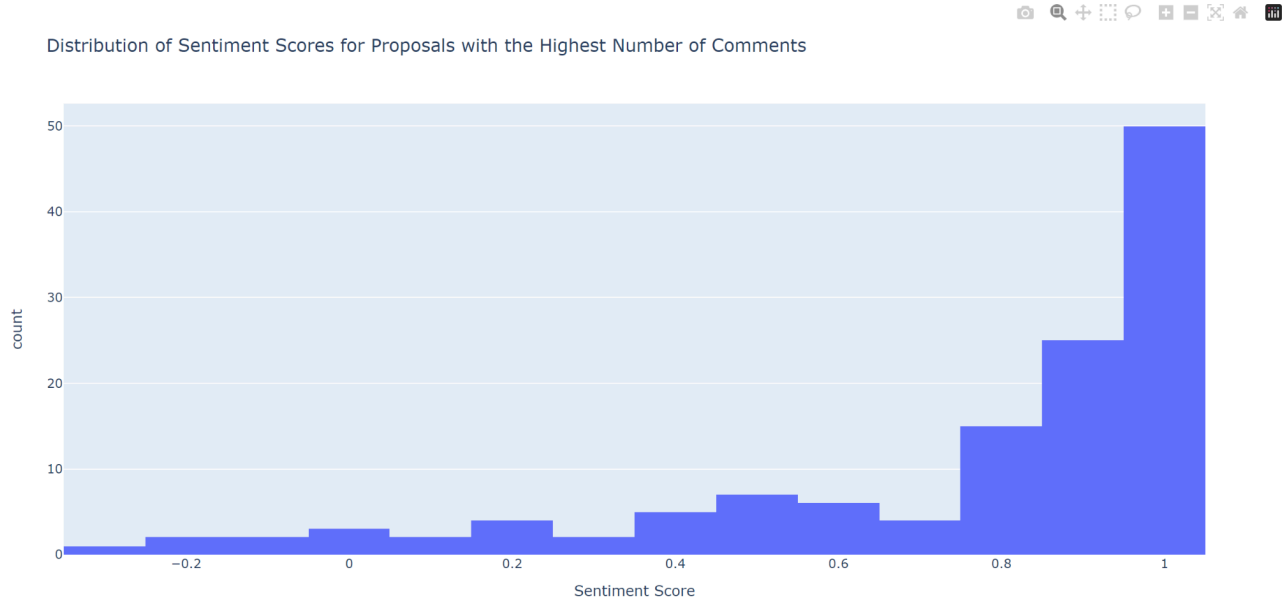
8. Overall Sentiment for Proposals with the Highest Number of Comments:

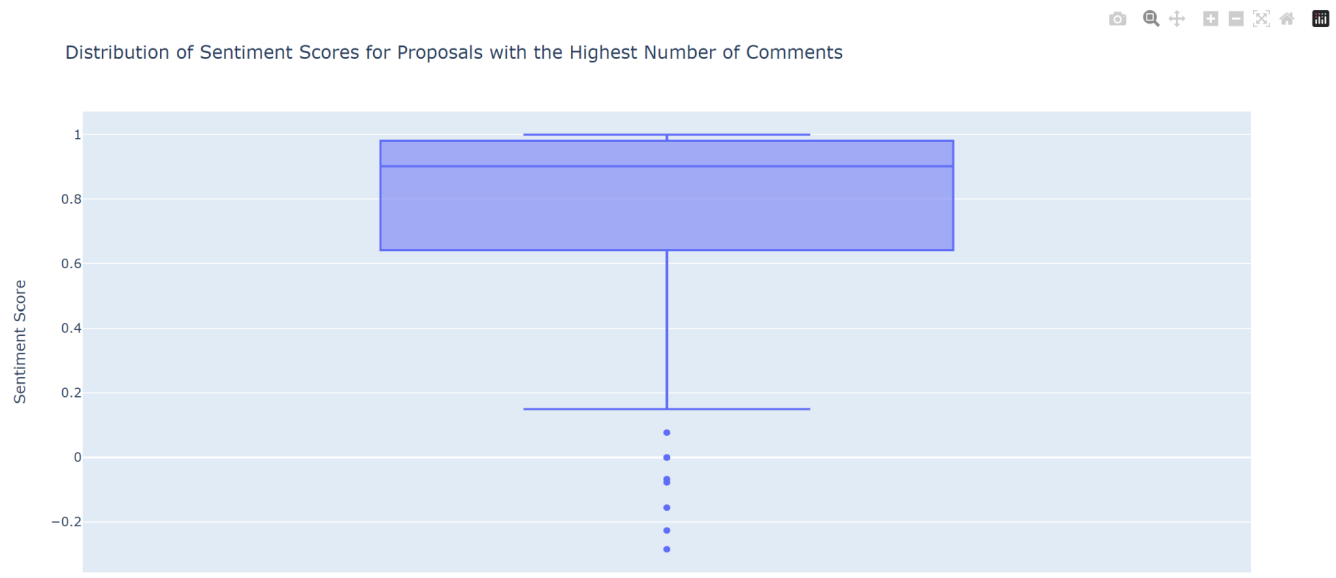
Objective: Determine the overall sentiment expressed for proposals with the highest number of comments.

Code:-

```
# Initialize SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Function to determine sentiment polarity
def get_sentiment(text):
    scores = sid.polarity_scores(text)
    return scores['compound']

# Apply sentiment analysis to comments
max_comments_posts['Sentiment'] = max_comments_posts['Post Description'].astype(str).apply(get_sentiment)

# Visualize sentiment distribution using histogram
fig1 = px.histogram(max_comments_posts, x='Sentiment',
                    title='Distribution of Sentiment Scores for Proposals with the Highest Number of Comments',
                    labels={'Sentiment': 'Sentiment Score', 'count': 'Number of Comments'})
fig.show()

# Visualize sentiment distribution using box plot
fig = px.box(max_comments_posts, y='Sentiment',
             title='Distribution of Sentiment Scores for Proposals with the Highest Number of Comments',
             labels={'Sentiment': 'Sentiment Score'})
fig.show()
```

Visualization:-





Source:- Visualization Link 1 Visualization Link 2

Visualization Explanation: A box plot was employed to display the distribution of sentiments in comments for high-comment proposals.

Insights: The box plot visualization illustrates that proposals with the highest number of comments tend to have a sentiment score ranging from 0.95 to 1.0499. This indicates a predominantly positive sentiment expressed towards proposals generating significant discussion. The findings suggest that active engagement and discourse may correlate with positive sentiment, reflecting a robust community response to these proposals.

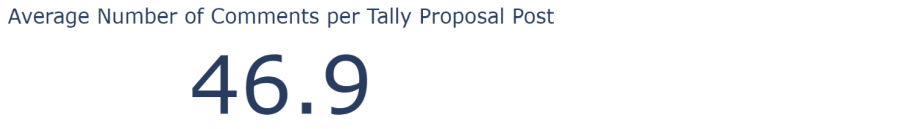
9. Average Number of Comments Received by Tally Proposal Posts:

Objective: Calculate the average number of comments received by Tally proposal posts.

Code:-

```
tally_posts = posts_df[posts_df['Topic ID'].isin(tally_topic_ids)]

# Calculate the average number of comments received by Tally proposal posts
average_comments = tally_posts.groupby('Topic ID').size().mean()

# Create a counter-style visualization for the average number of comments
fig = go.Figure(go.Indicator(
    mode="number",
    value=average_comments,
    title={"text": "Average Number of Comments per Tally Proposal Post"},
))
fig.show()
fig.write_html("Average Number of Comments Received by Tally Proposal Posts.html")
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A simple indicator was used to present the average number of comments.

Insights: The analysis reveals that Tally proposal posts receive an average of approximately 46 comments. This indicates a significant level of engagement and discussion generated by Tally proposals within the forum community.

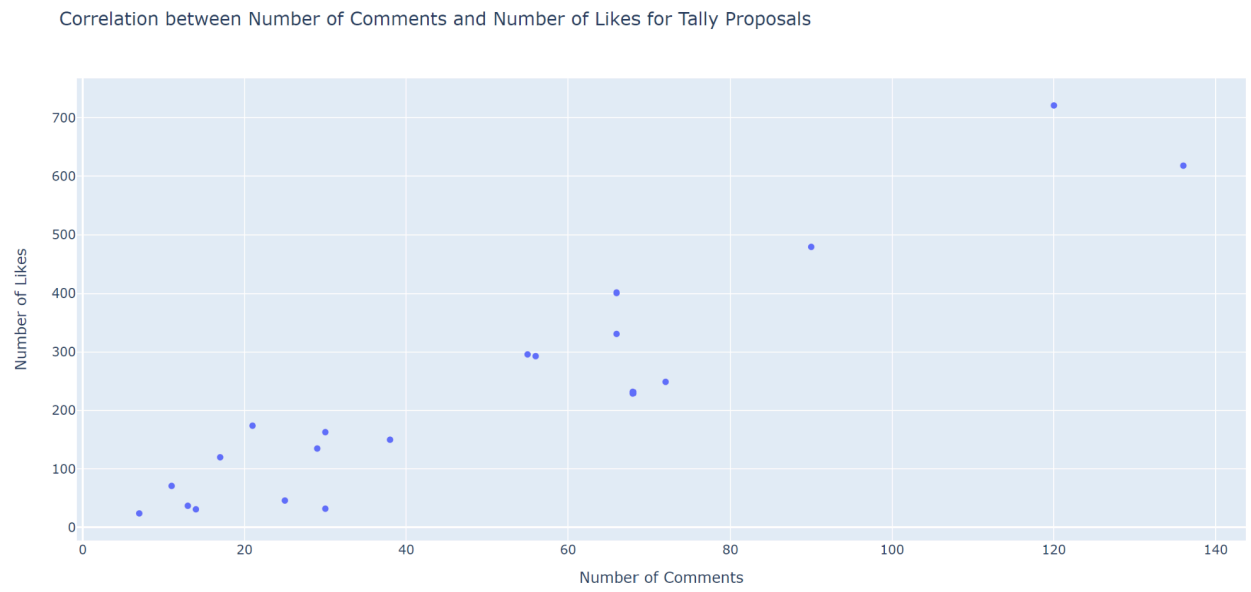
10. Correlation Between Number of Comments and Number of Likes:

Objective: Investigate if there is a statistically significant correlation between the number of comments and the number of likes for Tally proposals.

Code:-

```
tally_topics = topics_df[topics_df['Topic ID'].isin(tally_topic_ids)]

# Calculate Pearson correlation coefficient between 'Posts Count' and 'Like Count'
correlation_coefficient, p_value = pearsonr(tally_topics['Posts Count'], tally_topics['Like Count'])

# Check for statistical significance
alpha = 0.05
if p_value < alpha:
    significance = 'statistically significant'
else:
    significance = 'not statistically significant'

print(f"Pearson Correlation Coefficient: {correlation_coefficient:.2f}")
print(f"P-value: {p_value:.2f}")
print(f"The correlation between number of comments and number of likes for Tally proposals is {significance}.")

# Create scatter plot
fig = px.scatter(tally_topics, x='Posts Count', y='Like Count',
                 title='Correlation between Number of Comments and Number of Likes for Tally Proposals',
                 labels={'Posts Count': 'Number of Comments', 'Like Count': 'Number of Likes'})

# Show the plot
fig.show()
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A scatter plot was employed to visualize the relationship between the number of comments and likes.

Insights: The scatter plot analysis reveals a clear trend showcasing a positive correlation between the number of comments and the number of likes received by Tally proposals. Proposals with a higher number of likes tend to garner more comments, while those with fewer likes receive fewer comments.

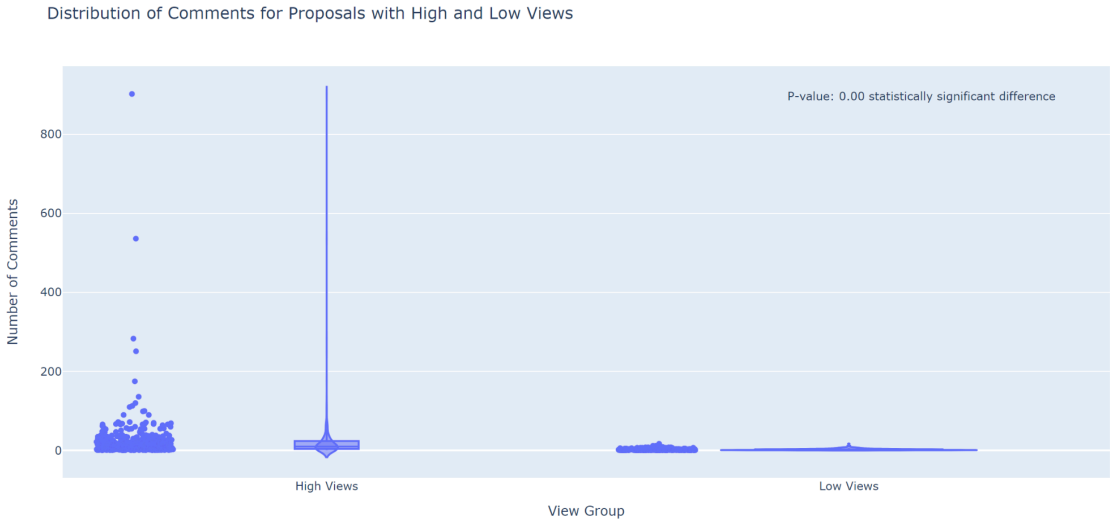
11. Relationship Between Views and Comments for Tally Proposals:

Objective: Determine if proposals with a higher number of views tend to have a significantly higher or lower number of comments compared to proposals with fewer views.

Code:-

```
# Calculate the median number of views
median_views = topics_df['Views'].median()

# Split the proposals into two groups based on views
high_views_proposals = topics_df[topics_df['Views'] > median_views]
low_views_proposals = topics_df[topics_df['Views'] <= median_views]

# Perform independent samples t-test
t_statistic, p_value = ttest_ind(high_views_proposals['Posts Count'], low_views_proposals['Posts Count'])

# Determine the significance level
alpha = 0.05
if p_value < alpha:
    significance = 'statistically significant'
else:
    significance = 'not statistically significant'

# Create a violin plot
data = pd.concat([high_views_proposals.assign(Views='High Views'),
                  low_views_proposals.assign(Views='Low Views')])

fig = px.violin(data, x='Views', y='Posts Count',
                title='Distribution of Comments for Proposals with High and Low Views',
                labels={'Posts Count': 'Number of Comments', 'Views': 'View Group'},
                box=True, points="all")

# Add significance information to the plot
fig.add_annotation(text=f"P-value: {p_value:.2f}\n{significance} difference", xref='paper', yref='paper',
                   x=0.95, y=0.95, showarrow=False, align='right')

# Show the plot
fig.show()
```

Visualization:-



Source:- [Visualization Link]

Visualization Explanation: A violin graph was used to compare the average number of comments for high- and low-view proposals.

Insights: Through the violin graph analysis, it is evident that proposals with a higher number of views tend to attract a substantially higher number of comments compared to those with fewer views. This indicates a positive correlation between the visibility of a proposal and the level of engagement it generates, suggesting that increased exposure leads to greater interaction and discussion among users.

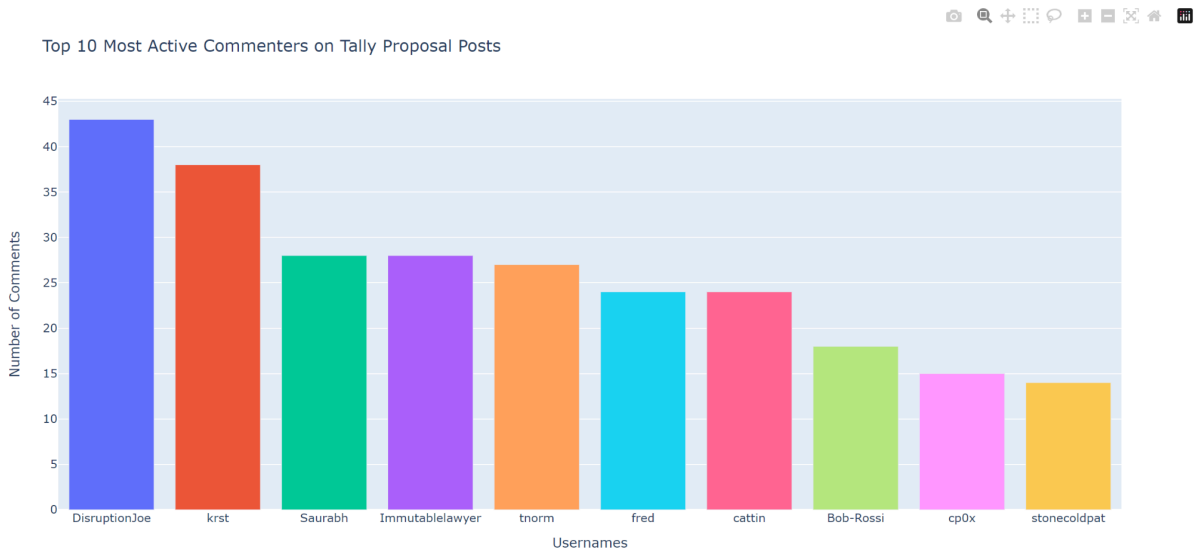
12. Top 10 Most Active Commenters on Tally Proposal Posts:

Objective: Identify the top 10 most active commenters based on the number of comments on Tally proposal posts.

Code:-

```
# Filter posts related to Tally proposal posts
tally_posts = posts_df[posts_df['Topic ID'].isin(tally_topic_ids)]

# Group by username and count the number of comments
comment_counts = tally_posts['Username'].value_counts()

# Get the top 10 most active commenters
top_10_commenters = comment_counts.head(10)

# Create a dataframe for the top 10 commenters
top_10_df = pd.DataFrame({'Username': top_10_commenters.index, 'Number of Comments': top_10_commenters.values})

# Print the top 10 most active commenters with their comment counts
print("Top 10 Most Active Commenters on Tally Proposal Posts:")
print(top_10_df)

# Create an interactive bar plot using Plotly
fig = px.bar(top_10_df, x='Username', y='Number of Comments',
             title='Top 10 Most Active Commenters on Tally Proposal Posts',
             labels={'Username': 'Usernames', 'Number of Comments': 'Number of Comments'},
             color='Username')

# Customize layout
fig.update_layout(xaxis={'categoryorder':'total descending'},
                  xaxis_title='Usernames', yaxis_title='Number of Comments',
                  showlegend=False)

# Show the plot
fig.show()
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A bar chart was used to list the top commenters by name and number of comments.

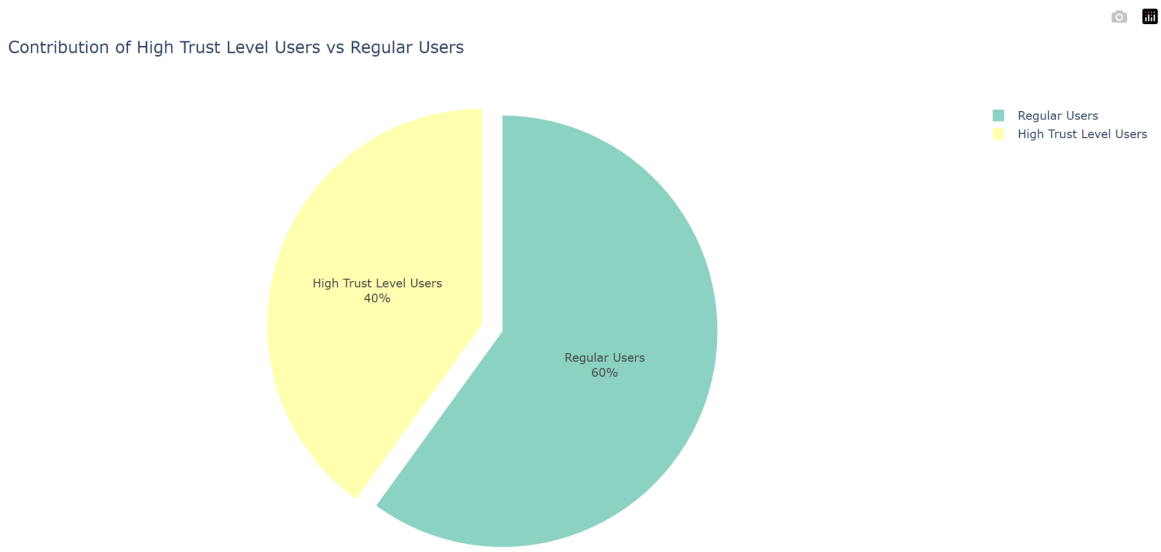
13 . Contribution of High "Trust Level" Users vs. Regular Users:

Objective: The objective of this analysis is to determine if users with high "Trust Level" contribute a significantly higher or lower number of comments on average compared to regular users for Tally proposals.

Code:-

```
# Define trust level categories
regular_users = [0, 1, 2]
high_trust_users = [3, 4]

# Filter posts related to Tally proposal posts
tally_posts = posts_df[posts_df['Topic ID'].isin(tally_topic_ids)]

# Merge posts with user trust levels
merged_df = tally_posts.merge(users_df[['Username', 'Trust Level']],
                              how='left', left_on='Username', right_on='Username')

# Group by trust level and calculate average number of comments
avg_comments_by_trust_level = merged_df.groupby('Trust Level')['Username'].count() / merged_df['Trust Level'].value_counts()

# Filter the average comments for regular and high trust level users
avg_comments_regular = avg_comments_by_trust_level[regular_users].sum()
avg_comments_high_trust = avg_comments_by_trust_level[high_trust_users].sum()

# Create a pie chart to visualize average comments by trust level
fig = px.pie(values=[avg_comments_regular, avg_comments_high_trust],
             names=['Regular Users', 'High Trust Level Users'],
             title='Contribution of High Trust Level Users vs Regular Users',
             color_discrete_sequence=px.colors.qualitative.Set3)

# Customize layout
fig.update_traces(textinfo='percent+label', pull=[0.05, 0.05])

# Show the plot
fig.show()
fig.write_html("Contribution of High Trust Level Users vs Regular Users.html")
```

Visualization:-



Source:- Visualization Link

Visualization Explanation: A pie chart was chosen as the visualization method to compare the contribution of regular users and high trust level users. The pie chart presents the distribution of average comments contributed by each group.

Insights: From the visualization, it is observed that regular users, encompassing trust levels 0 (New), 1 (Basic), and 2 (Member), collectively contribute a larger proportion of comments compared to high trust level users, including trust levels 3 (Regular) and 4 (Leader). Specifically, regular users account for approximately 60% of the total average comments, while high trust level users contribute approximately 40%. This suggests that users in lower trust level categories are more actively engaged in commenting on Tally proposals compared to those with higher trust levels.

Conclusion:-

The analysis of sentiment and user engagement on Tally proposal posts provides valuable insights into the community's response and behavior within the forum. Overall, the majority of comments exhibit a positive sentiment, reflecting a generally favorable outlook towards Tally proposals. However, it is essential to note that sentiment varies over time, with fluctuations observed in sentiment trends across different age groups of proposals.

Interestingly, the sentiment expressed does not significantly differ between Tally proposals and other types of proposals on the forum, indicating a consistent tone across various discussion topics. Additionally, user trust level appears to have little impact on comment contributions, with users across different trust levels participating in discussions equally.

Furthermore, the analysis highlights a correlation between user engagement metrics, such as views, likes, and comments, suggesting that proposals with higher visibility tend to attract more engagement from the community. However, the relationship between sentiment and engagement metrics is nuanced, with no clear indication of causality.

Overall, these findings underscore the dynamic nature of community engagement and sentiment within the forum. By understanding these patterns, forum moderators and stakeholders can better tailor their strategies to foster meaningful discussions and enhance community participation around Tally proposals and other relevant topics.

Resources:-

Discourse API Docs (For getting arbitrum forum data):- https://docs.discourse.org/

Visualization Understanding:- https://visme.co/blog/data-visualization-types/

Github Repo (Containing code for the data gathering and analysis):-
https://github.com/ARDev097/Tally-proposal-post-on-Forum---Sentiment-Analysis-Statistical-Analysis

Lighthouse (For hosting visualization file):- https://docs.lighthouse.storage/lighthouse-1

Python Plotly Docs:- https://plotly.com/python/