Fakebook Marketplace Case Project

Yashmith Raj
SRM AP University

Data Preprocessing: Handling Missing Data

- From the output, we can see that Column1, Column2, Column3, and Column4 are completely empty, so we can drop them to clean the dataset.

```
df = df.drop(columns = ['Column1','Column2','Column3','Column4'])
print(df.columns)
```

```
Index(['status_id', 'status_type', 'status_published', 'num_reactions',
       'num_comments', 'num_shares', 'num_likes', 'num_loves', 'num_wows',
       'num_hahas', 'num_sads', 'num_angrys'],
      dtype='object')
```

Handling Missing Values

```
print(df.isnull().sum())
```

```
status_id           0
status_type         0
status_published    0
num_reactions       0
num_comments        0
num_shares          0
num_likes           0
num_loves           0
num_wows            0
num_hahas           0
num_sads            0
num_angrys          0
Column1          7050
Column2          7050
Column3          7050
Column4          7050
dtype: int64
```

Data Preprocessing: Handling Categorical Data

- From the output, we can see that status_type and status_published are listed as categorical data.

- The status_type column has four categories: photo, video, status, and link, with photo being the most frequent.
- Applied Label Encoding to the status_type column to convert it into numerical format for modeling.

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7050 entries, 0 to 7049
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   status_id         7050 non-null   int64
 1   status_type       7050 non-null   object
 2   status_published  7050 non-null   object
 3   num_reactions     7050 non-null   int64
 4   num_comments      7050 non-null   int64
 5   num_shares        7050 non-null   int64
 6   num_likes         7050 non-null   int64
 7   num_loves         7050 non-null   int64
 8   num_wows          7050 non-null   int64
 9   num_hahas         7050 non-null   int64
 10  num_sads          7050 non-null   int64
 11  num_angrys        7050 non-null   int64
dtypes: int64(10), object(2)
```

```
print(df['status_type'].value_counts())
```

```
status_type
photo     4288
video     2334
status     365
link        63
Name: count, dtype: int64
```

Data Preprocessing: Handling Categorical Data

- From the previous output, we can see that status_published is listed as categorical data.
- The status_published column seems to represent date and time.
- We can convert the status_published column from string format to datetime format using the pandas to_datetime function for easier manipulation and analysis.

```
print(df['status_published'].describe())
```

```
count                            7050
mean     2016-11-20 05:13:52.672340224
min                2012-07-15 02:51:00
25%                2016-03-15 16:52:45
50%                2017-11-18 01:19:00
75%                2018-03-09 05:26:45
max                2018-06-13 01:12:00
Name: status_published, dtype: object
```

```
print(df['status_published'].head())
```

```
0     4/22/2018 6:00
1    4/21/2018 22:45
2     4/21/2018 6:17
3     4/21/2018 2:29
4     4/18/2018 3:22
Name: status_published, dtype: object
```

```
df['status_published'] = pd.to_datetime(df['status_published'], format='%m/%d/%Y %H:%M')
print(df['status_published'].head())
```

```
0   2018-04-22 06:00:00
1   2018-04-21 22:45:00
2   2018-04-21 06:17:00
3   2018-04-21 02:29:00
4   2018-04-18 03:22:00
Name: status_published, dtype: datetime64[ns]
```

1. How does the time of upload status_published affects the `num_reaction`?

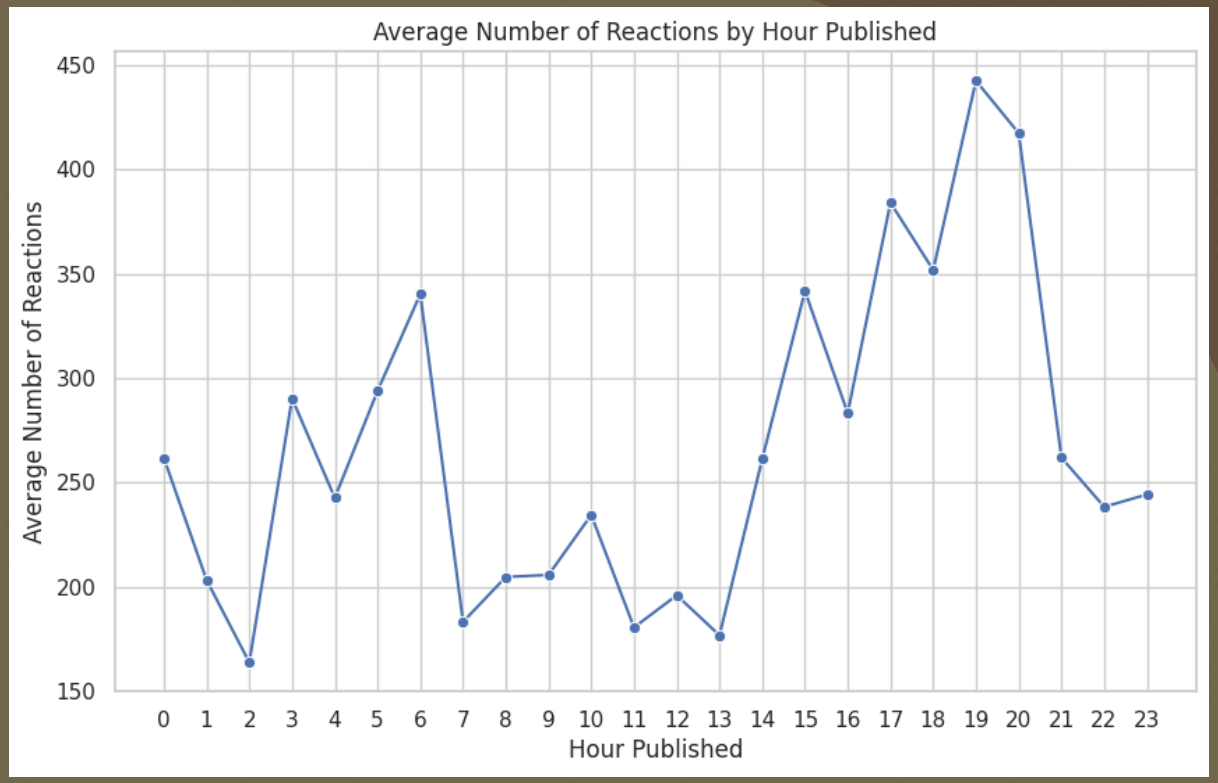
- To determine how the time of upload status_published impacts the number of reactions (num_reactions).
- By extracting the hour, day, and month from the status_published column using functions like dt.hour, dt.dayofweek, and dt.month, we can observe how these time factors influence the average number of reactions. This analysis helps identify the optimal times for posting to maximize engagement.
- By grouping and calculating the average reactions for different time factors, we can observe patterns in how the timing of posts affects engagement.

```
df['hrs_pub'] = df['status_published'].dt.hour
df['day_pub'] = df['status_published'].dt.dayofweek
df['month_pub'] = df['status_published'].dt.month
print(df.head())
```

```
hrs_rec = df.groupby('hrs_pub')['num_reactions'].mean().reset_index()
day_rec = df.groupby('day_pub')['num_reactions'].mean().reset_index()
month_rec = df.groupby('month_pub')['num_reactions'].mean().reset_index()
print(hrs_rec)
```

Reactions Based on Hours

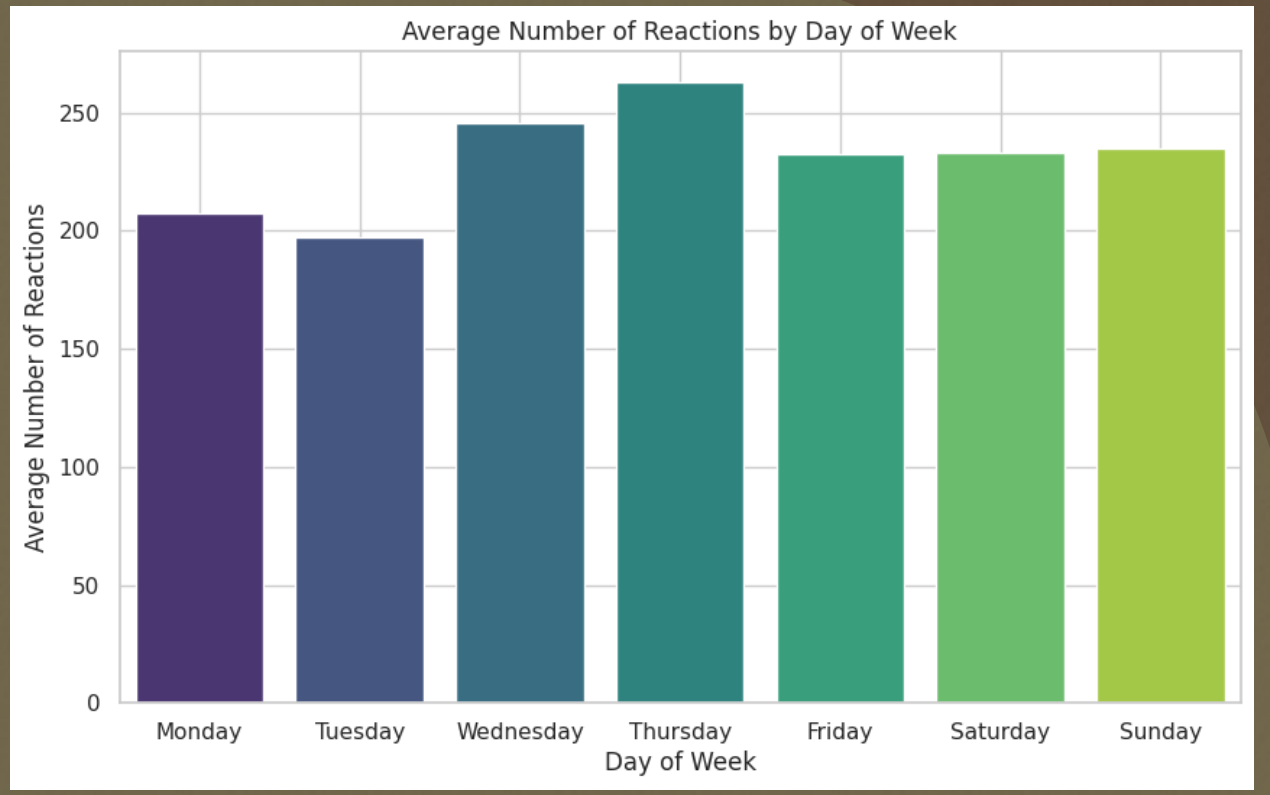
- **Peak Hours:**
  - The highest average number of reactions is observed around 19:00 (7 PM).
  - Notable peaks also occur at 17:00 (5 PM) and 18:00 (6 PM).
- **Low Activity Hours:**
  - Early morning hours, specifically around 03:00 (3 AM) and 13:00 (1 PM), exhibit the lowest average number of reactions.
  - A drop in reactions is also noted around 07:00 (7 AM).
- **Steady Hours:**
  - Reactions remain relatively steady from approximately 09:00 (9 AM) to 12:00 (12 PM), with only slight variations.



- **Conclusion:**
  - Posting in the evening, particularly between 17:00 and 20:00, tends to generate higher engagement.
  - Posts made in the early morning hours typically receive fewer reactions.

Reactions Based on Day

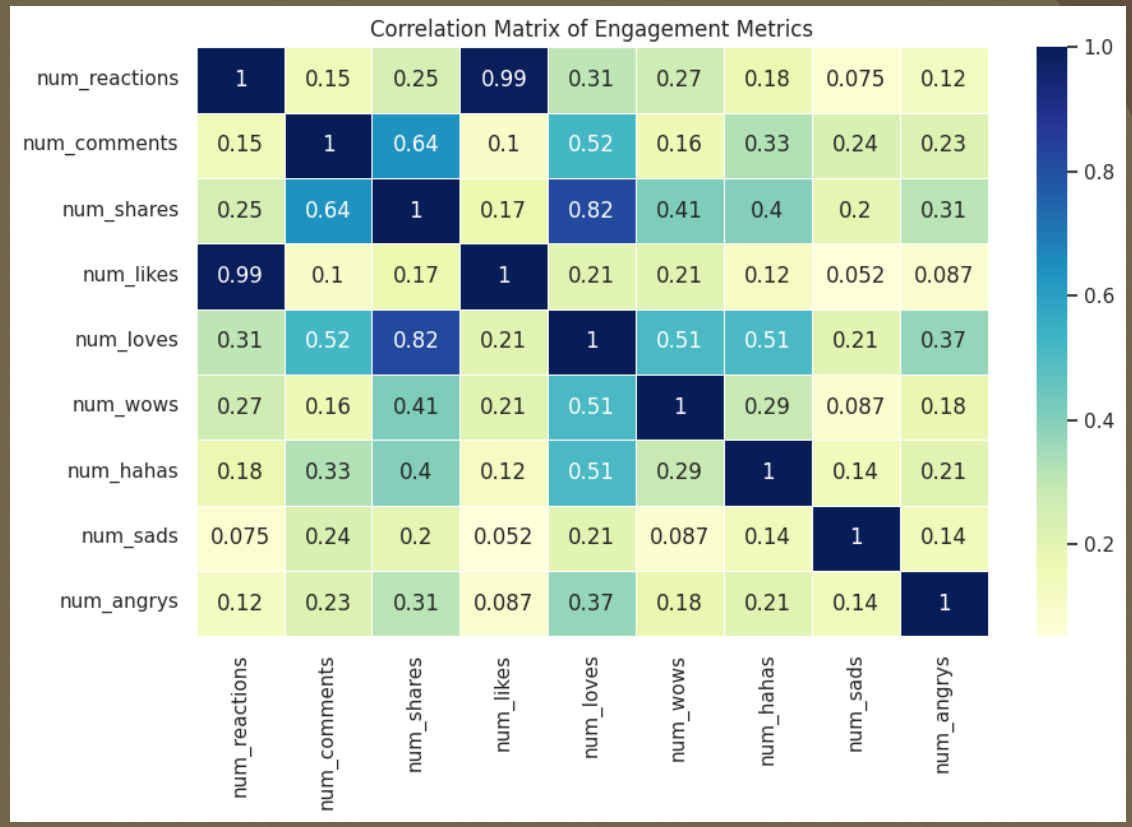
- **Highest Engagement:**
  - **Thursday** records the highest average number of reactions.
  - **Wednesday**, **Saturday**, and **Sunday** also exhibit high engagement levels, though slightly below Thursday.
- **Moderate Engagement:**
  - **Monday** and **Friday** have moderate engagement, with Monday being slightly higher than Tuesday, and Friday slightly lower than Thursday.
- **Lower Engagement:**
  - **Tuesday** shows the lowest average number of reactions.



- **Conclusion:**
  - Posting on **Thursday**, **Wednesday**, **Saturday**, and **Sunday** generally results in higher engagement.
  - **Tuesday** appears to be the least effective day for generating reactions.

2. Is there a correlation between the number of reactions and other engagement metrics

- Based on the correlation matrix:
- Number of Likes (num_likes):
  - Correlation: 0.99 (Very High)
  - Insight: Likes are the primary driver of total reactions. An increase in likes corresponds strongly with an increase in the total number of reactions.
- Number of Loves (num_loves):
  - Correlation: 0.31 (Moderate)
  - Insight: Posts that receive more loves tend to get more reactions overall.
- Number of Wows (num_wows):
  - Correlation: 0.27 (Moderate)
  - Insight: A moderate correlation indicates that posts with more wows also generally have more total reactions.
- Number of Comments (num_comments):
  - Correlation: 0.15 (Low)
  - Insight: The relationship between comments and total reactions is weak, showing that comments have a minor impact on overall reactions.
- Number of Shares (num_shares):
  - Correlation: 0.25 (Moderate)
  - Insight: Posts with more reactions are somewhat more likely to be shared, though the correlation is moderate.



Conclusion: Likes have the strongest correlation with the total number of reactions, while loves and wows also show moderate correlations. Comments and shares have weaker correlations with reactions.

3. Train a K-Means clustering model on the Facebook Live Sellers dataset using specified columns.

- Data Preprocessing for K-Means Clustering:
  - 1. Separating Columns:
    - To prepare the data for clustering, we first need to separate the categorical and numerical columns.
    - status_type is stored separately as categorical data cannot be used directly in clustering algorithms.

```
cat_clm = df[['status_type']]
num_clm = df[['num_reactions', 'num_comments', 'num_shares', 'num_like
```

3. Train a K-Means clustering model on the Facebook Live Sellers dataset using specified columns.

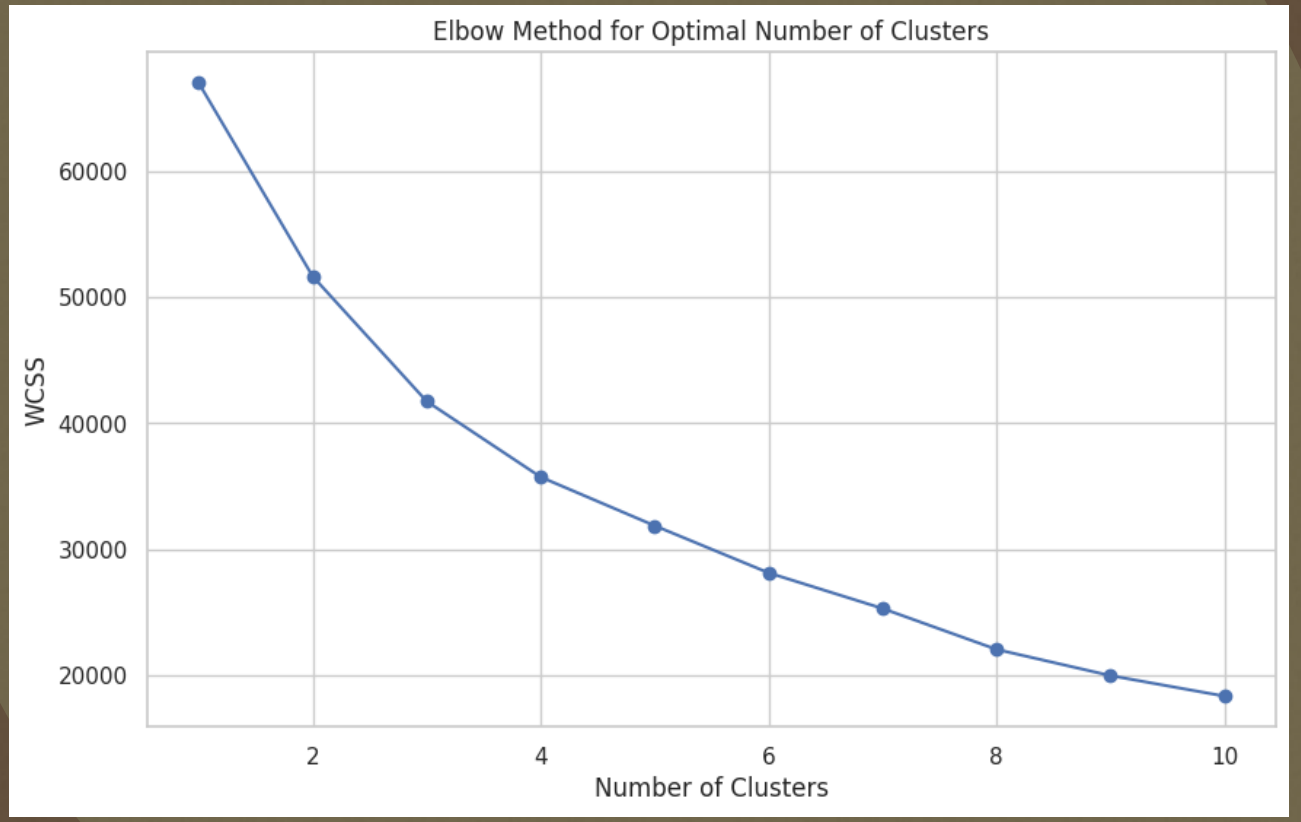
- 2. Preparing the Data for Clustering:
  - **One-Hot Encoding:** Converts the categorical status_type into numerical format using one-hot encoding. This creates binary columns for each category, facilitating its use in the clustering algorithm.
  - **Standardization:** Scales the numerical columns so that all features contribute equally to the clustering process. This ensures that no single feature disproportionately influences the clustering due to its scale.

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler

en = OneHotEncoder(sparse=False, drop='first')
cat_clm_en = en.fit_transform(cat_clm)

sc = StandardScaler()
num_clm_sc = sc.fit_transform(num_clm)

df_cat = pd.DataFrame(cat_clm_en, columns=en.get_feature_names_out())
df_num = pd.DataFrame(num_clm_sc, columns=num_clm.columns)

dfML = pd.concat([df_cat, df_num], axis=1)
print(dfML.head())
```

3. Train a K-Means clustering model on the Facebook Live Sellers dataset using specified columns.

❖3. Selecting the Number of Clusters (k):

- To determine the optimal number of clusters for the K-Means algorithm, we use the **Elbow Method**.
- This involves plotting the **Within-Cluster Sum of Squares (WCSS)** for different values of k and identifying the point where the WCSS starts to decrease at a slower rate, indicating the optimal number of clusters.

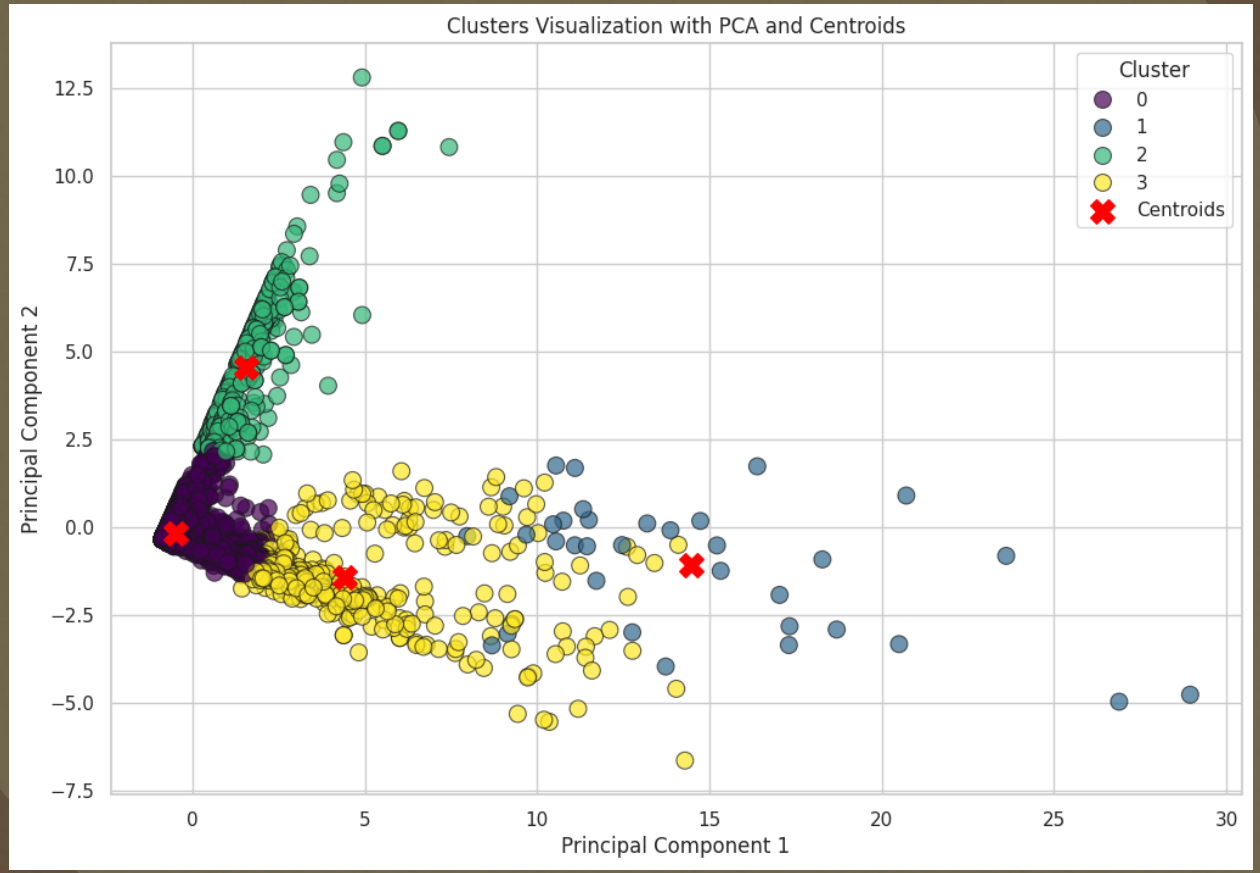


3. Train a K-Means clustering model on the Facebook Live Sellers dataset using specified columns.

- 4. Apply K-means Clustering to the standardized data.

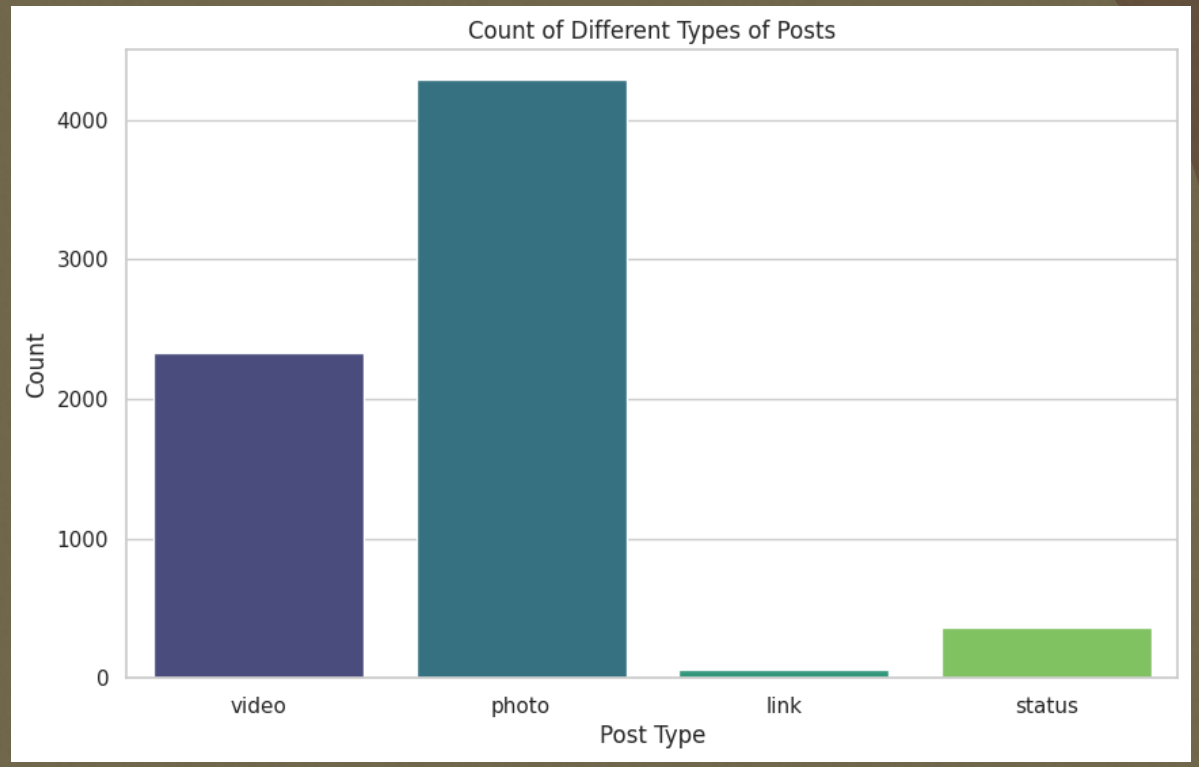
- 5. **Apply PCA for Visualization:**
  - Since you're likely dealing with multiple dimensions (engagement metrics) use PCA.
  - Principal Component Analysis (PCA) is used to reduce the dimensionality to 2 principal components (PC1 and PC2). These capture the most significant variance in the data, allowing for visualization in a 2D plot.

- **6.Plot the Clusters:**
  - Create a scatter plot of the clusters in the 2D PCA space, including cluster centroids.



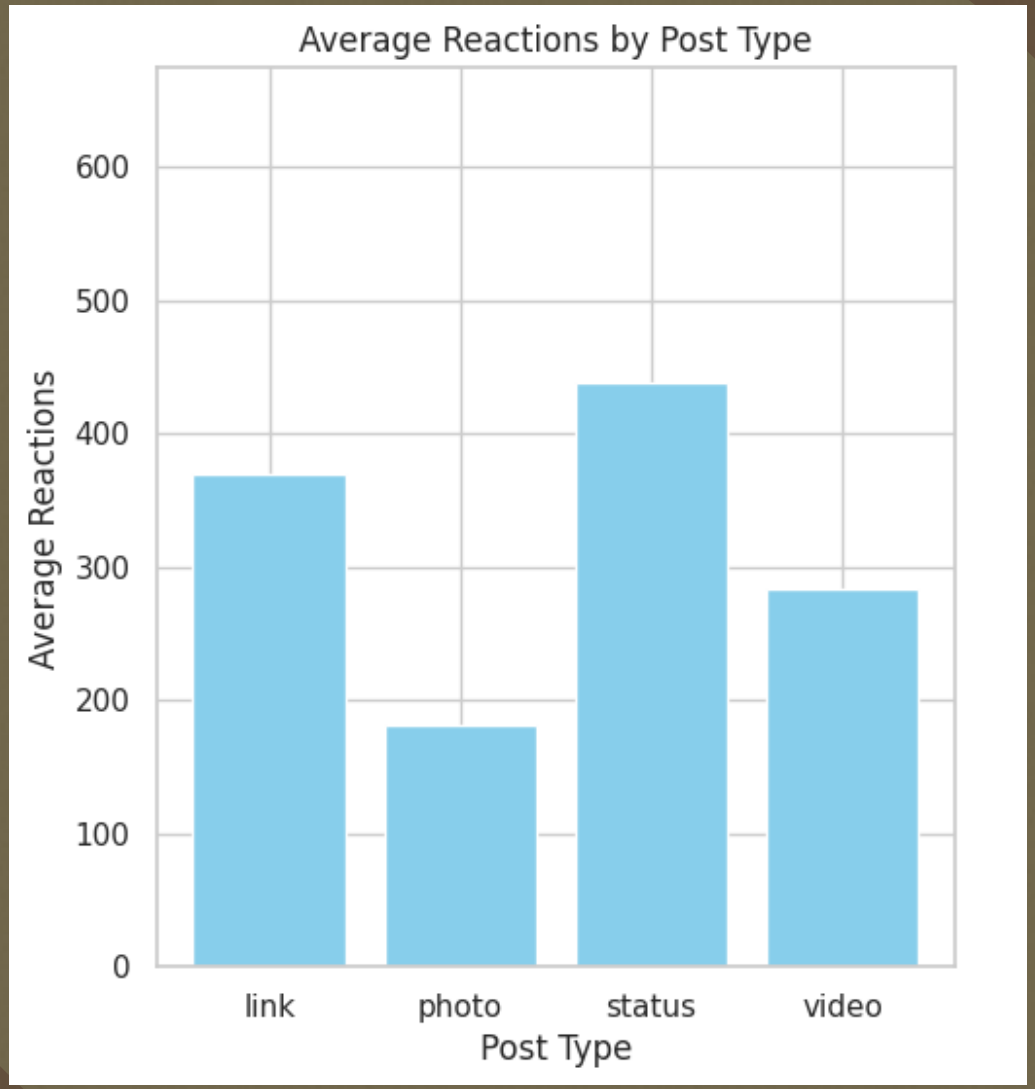
5. What is the count of different types of posts in the dataset?

- The dataset categorizes posts into four types: Photo, Video, Status, and Link.
- **Photo Posts:**
  - **Count:** 4,288
  - Photo posts are the most common, showing that visual content is heavily used to engage the audience.
- **Video Posts:**
  - **Count:** 2,334
  - Videos are the second most common. They are used to capture attention and provide detailed content.
- **Status Posts:**
  - **Count:** 365
  - Text-only status updates are less frequent, suggesting they are not as effective for engagement as photos and videos.
- **Link Posts:**
  - **Count:** 63
  - Posts with links are the least common, indicating a preference for keeping users on their own pages rather than sending them elsewhere.



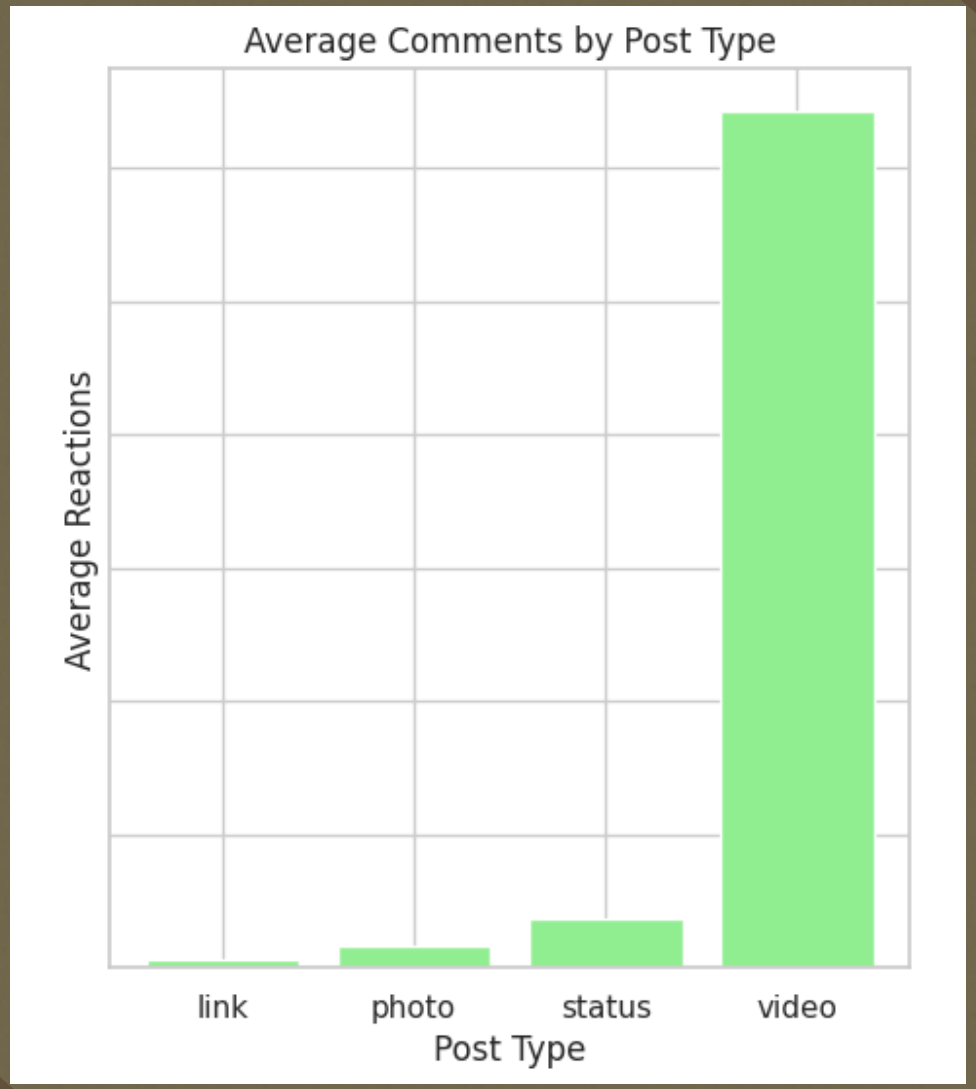
6. What is the average value of num_reaction, num_comments, num_shares for each post type?

- **Average Reactions by Post Type:**
  - **Link Posts:** 370.14
  - **Photo Posts:** 181.29
  - **Status Posts:** 438.78
  - **Video Posts:** 283.41
- Status posts have the highest average reactions, while photo posts have the lowest.



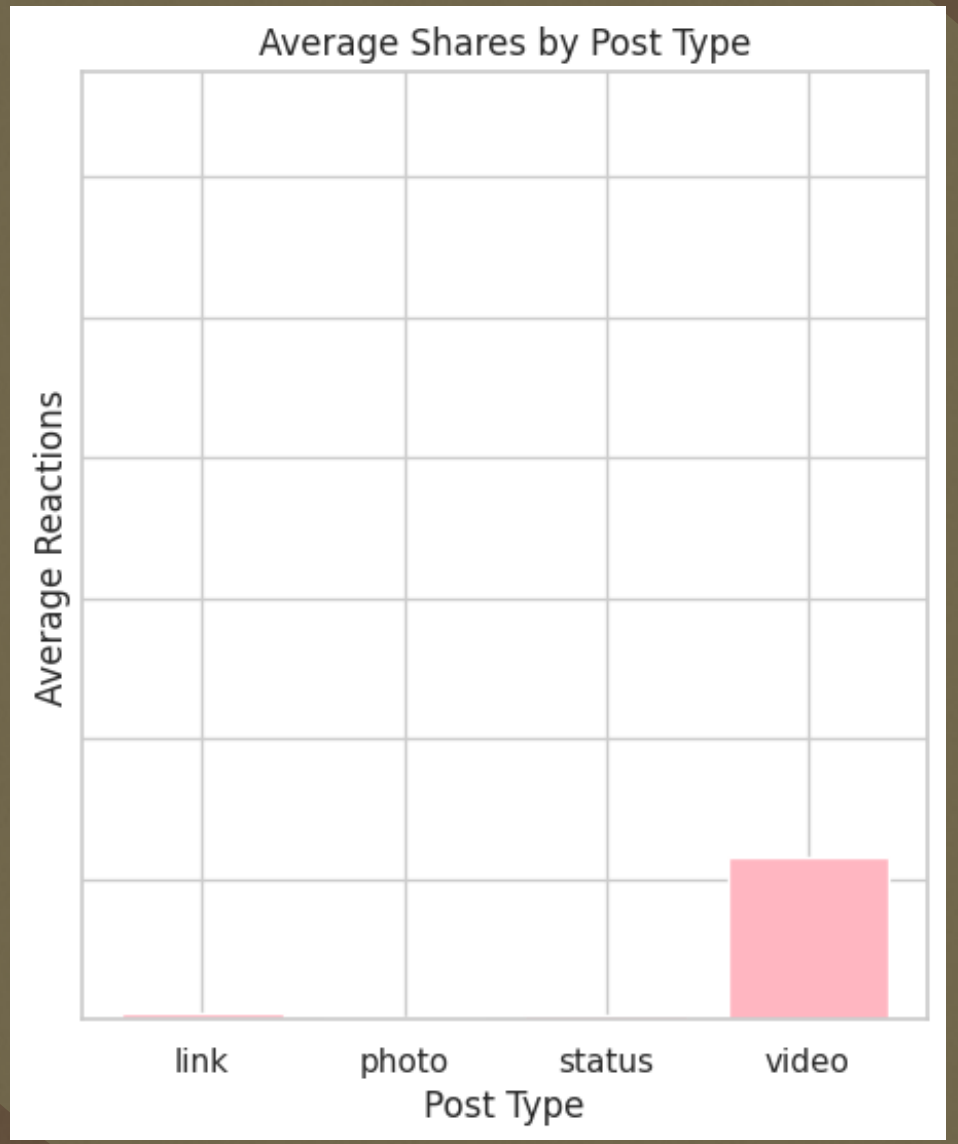
6. What is the average value of num_reaction, num_comments, num_shares for each post type?

- **Average Comments by Post Type**
  - **Link Posts:** 5.70
  - **Photo Posts:** 15.99
  - **Status Posts:** 36.24
  - **Video Posts:** 642.48
- Video posts receive the most comments on average, whereas link posts receive the fewest.



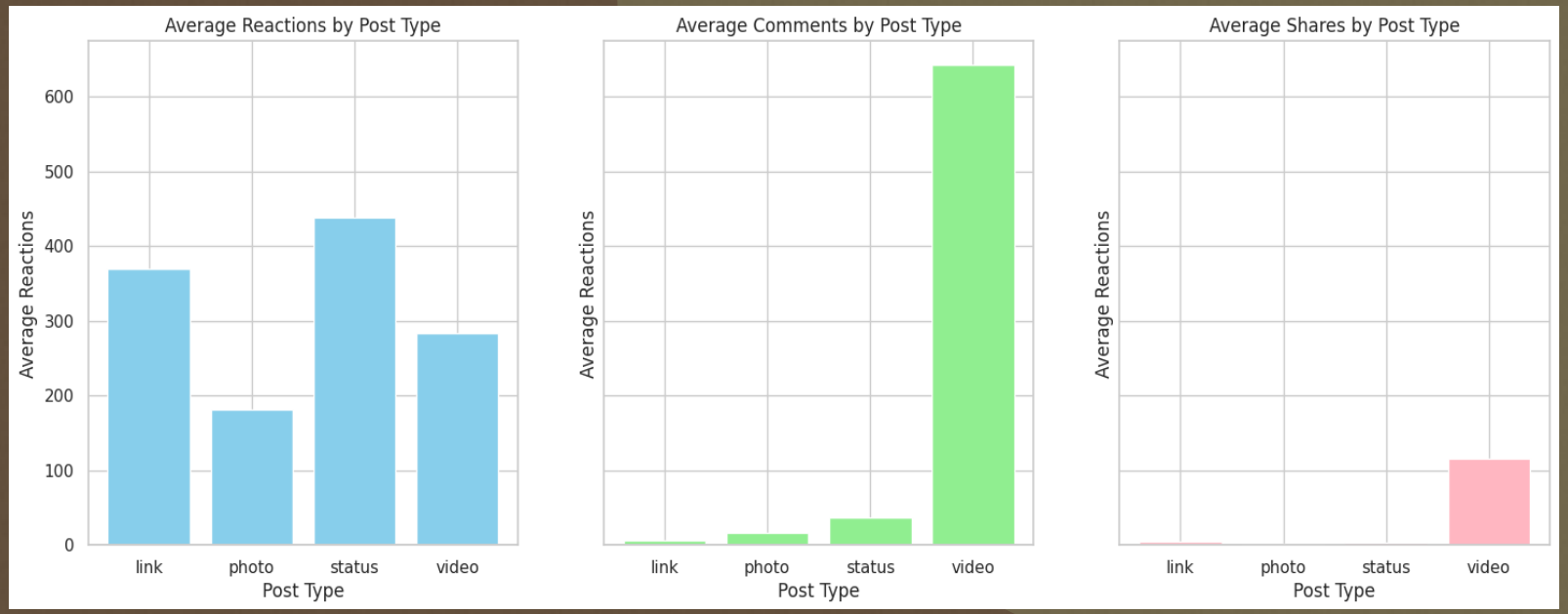
6. What is the average value of num_reaction, num_comments, num_shares for each post type?

- **Average Shares by Post Type**
  - **Link Posts:** 4.40
  - **Photo Posts:** 2.55
  - **Status Posts:** 2.56
  - **Video Posts:** 115.68
- Video posts are shared the most, while photo and status posts have lower average shares.



6. What is the average value of num_reaction, num_comments, num_shares for each post type?

- **Conclusion:**
  - **Video Posts** stand out for high engagement, particularly in comments and shares.
  - **Status Posts** have high average reactions but fewer shares and comments.
  - **Photo Posts** have moderate engagement, especially in reactions and comments.
  - **Link Posts** generally show lower engagement metrics compared to other types.



My Experience:

- **Principal Component Analysis (PCA):**
  - Gained insights into using PCA for reducing dimensionality and visualizing high-dimensional data. This technique helped simplify the visualization of clusters by projecting data onto the first two principal components, making it easier to interpret clustering results.
- **Using the Elbow Method (WCSS):**
  - Applied the Elbow Method to determine the optimal number of clusters for the K-Means algorithm. By evaluating the Within-Cluster Sum of Squares (WCSS), I was able to identify the most appropriate number of clusters, improving the effectiveness of the clustering analysis.
- **Handling Categorical Data for K-Means Clustering:**
  - Understood the process of converting categorical data into numerical format using One-Hot Encoding. This step was crucial for enabling the K-Means algorithm to accurately process and cluster different post types based on their engagement metrics.
- **Cluster Interpretation and Visualization:**
  - Developed skills in interpreting cluster distribution and centroid placement in the context of engagement metrics. This involved visualizing clusters with PCA and understanding how different types of posts are grouped based on their engagement characteristics.