Experiment No.3
To perform data cleaning on social media data using python or R.
Date of Performance:28/01/2025
Date of Submission:04/02/2025

Aim : To perform data cleaning on social media data using python.

Objective : To analyze data cleaning on social media data using Python is to prepare the data for meaningful analysis, ensure data integrity and quality, and facilitate efficient and ethical use of the data for generating actionable insights.

Theory :
Data cleaning is a crucial step in the data preprocessing pipeline. It involves identifying and correcting errors, inconsistencies, and inaccuracies in the data to improve its quality and reliability. In the context of social media data, which is often unstructured and noisy, data cleaning becomes even more essential.

- **Ensure Data Quality:** The primary objective of data cleaning is to ensure that the data is accurate, consistent, and reliable. Social media data can contain various types of errors such as misspellings, grammatical mistakes, and inconsistencies that need to be addressed.
- **Handle Missing Values:** Social media data often contains missing values due to incomplete user inputs or data collection processes. Data cleaning involves identifying and handling these missing values appropriately, either by imputation or removal.
- **Remove Duplicates:** Social media data may contain duplicate entries, such as duplicate posts or comments. Removing duplicates ensures that each piece of information is unique and prevents redundancy in the dataset.
- **Standardize Formats:** Social media data can have diverse formats for representing dates, times, and other structured information. Data cleaning involves standardizing these formats to facilitate analysis and comparison across different data points.
- **Text Cleaning and Preprocessing:** Since social media data often consists of text data, cleaning and preprocessing text is essential. This may include removing special characters, URLs, hashtags, mentions, and other noise, as well as tokenization, lemmatization, and removing stopwords to prepare the text for analysis.
- **Ensure Consistency and Uniformity:** Data cleaning ensures that the data is consistent and uniform across different attributes and records. This consistency is crucial for accurate analysis and modeling.
- **Enhance Analytical Results:** Clean data leads to more accurate and reliable analytical results. By removing errors and inconsistencies, data cleaning improves the quality of insights derived from social media data analysis.
- **Compliance and Ethical Considerations:** Data cleaning may also involve ensuring compliance with regulations such as GDPR (General Data Protection Regulation) and addressing ethical considerations such as privacy concerns when dealing with sensitive user data in social media datasets

Handle Missing Values: Check for missing values and decide how to handle them. Options include dropping rows with missing values, filling them with a default value, or using more sophisticated methods like interpolation.

```
# Drop rows with missing values
    data.dropna(inplace=True)
# Fill missing values with a default value
    data.fillna(0, inplace=True)
```

Remove Duplicates: Remove any duplicate rows in the dataset.

```
data.drop_duplicates(inplace=True)
```

Text Cleaning: Preprocess text data by removing special characters, URLs, hashtags, mentions, and performing other text cleaning tasks.

```
    def
clean_text(text): #
Remove URLs
    text = re.sub(r'http\S+', '', text)
# Remove special characters and punctuation
    text = re.sub(r'[^\w\s]', '', text)
# Remove numbers
    text = re.sub(r'\d+', '', text)
# Convert to lowercase
    text = text.lower()
    return text data['clean_text'] = data['text'].apply(clean_text)
```

Tokenization and Lemmatization/Stemming: Tokenize the text and perform lemmatization or stemming to standardize words.

```
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
        def
tokenize_and_lemmatize(text): tokens =
word_tokenize(text)
lemmatized_tokens = [lemmatizer.lemmatize(token) for token in tokens]
return lemmatized_tokens
data['tokenized_text'] = data['clean_text'].apply(tokenize_and_lemmatize)
```

Program:

```
# 1. Handle Missing Values
# Fill missing numerical values with the mean of their column
```

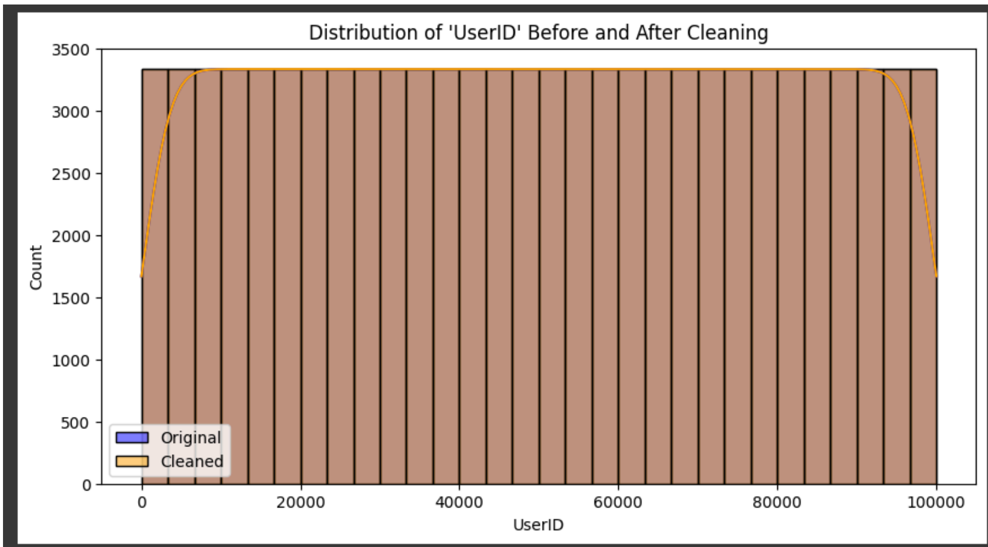
```
numerical_cols = data.select_dtypes(include=['float64',
'int64']).columns
for col in numerical_cols:
    if data[col].isnull().sum() > 0:
        data[col].fillna(data[col].mean(), inplace=True)
        print(f"Filled missing values in numerical column '{col}' with
mean: {data[col].mean()}")
# Show updated dataset after handling missing numerical values
print("Dataset after filling missing numerical values:")
print(data.head())
```

```
Dataset after filling missing numerical values:
   UserID                Name  Gender DOB  \
0       1       jesse lawhorn  female
1       2         stacy payne  female
2       3  katrina nicewander  female
3       4       eric yarbrough    male
4       5        daniel adkins  female

                                         Interests        City    Country
0                           movies fashion fashion books     sibolga  indonesia
1  gaming finance and investments outdoor activit...     al abyr      libya
2                  diy and crafts music science fashion  wd as sr     jordan
3              outdoor activities cars and automobiles      matera      italy
4                                    politics history    biruaca  venezuela
```

```
# 2. Data Cleaning
# Remove unnecessary columns (example: ID or empty columns)
unnecessary_columns = ['DOB']      # Remove 'ID' and 'dob' columns if
they exist
data.drop(columns=[col for col in unnecessary_columns if col in
data.columns], inplace=True)
# Show updated dataset after removing unnecessary columns
print("Dataset after removing unnecessary columns (including 'dob'):")
print(data.head(20))
```

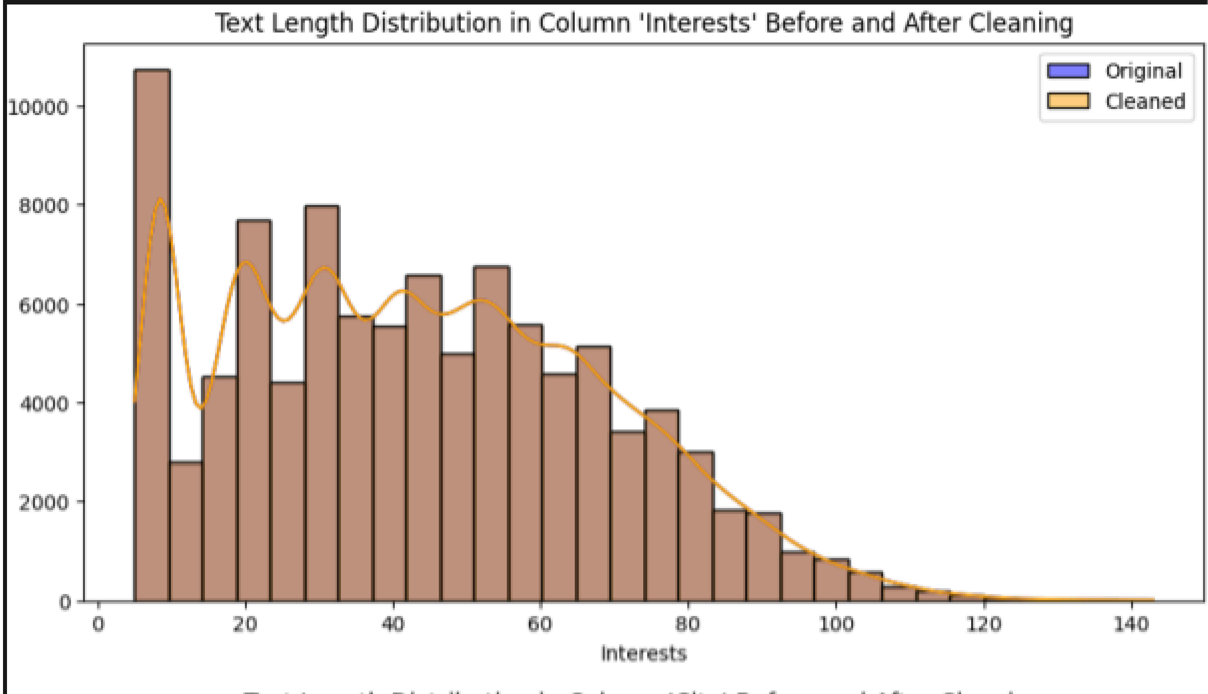
```
Dataset after removing unnecessary columns (including 'dob'):
    UserID                Name  Gender  \
0        1       jesse lawhorn  female
1        2         stacy payne  female
2        3  katrina nicewander  female
3        4      eric yarbrough    male
4        5       daniel adkins  female
5        6          diane jara    male
6        7        sheryl morgan  female
7        8      william harper    male
8        9     virginia varron    male
9       10    charles figueroa  female
10      11          paul chain    male
11      12    christina parker  female
12      13        jack freeman    male
13      14        wayne fasano    male
14      15       homer maxwell    male
15      16        frank holmes  female
16      17        donald zeller    male
17      18       sheldon wentz  female
18      19     isabel williams    male
19      20         cody watson  female

                                          Interests          City  \
0                      movies fashion fashion books       sibolga
1   gaming finance and investments outdoor activit...      al abyr
2               diy and crafts music science fashion       wd as sr
```

```python
# Plot distribution of a numerical column before and after cleaning
(example: first numerical column)
if not numerical_cols.empty:
    first_numerical_col = numerical_cols[0]
    plt.figure(figsize=(10, 5))
    sns.histplot(original_data[first_numerical_col].dropna(),
color='blue', label='Original', kde=True, bins=30)
    sns.histplot(data[first_numerical_col], color='orange',
label='Cleaned', kde=True, bins=30)
    plt.title(f"Distribution of '{first_numerical_col}' Before and
After Cleaning")
    plt.legend()
    plt.show()
```

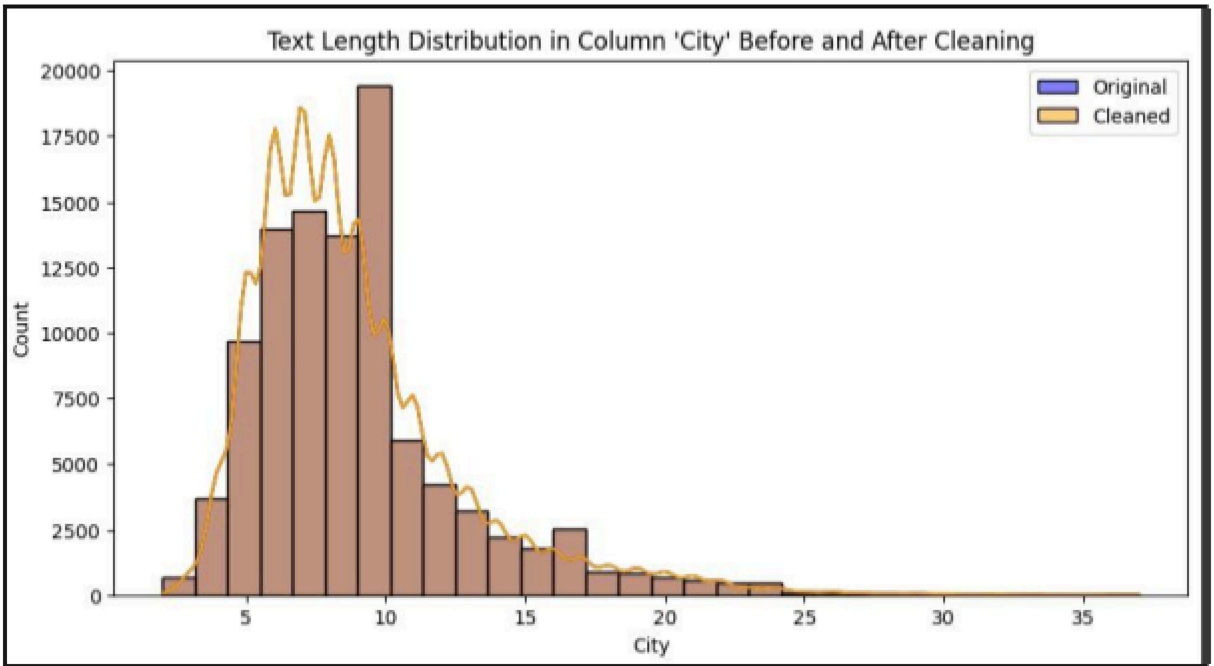


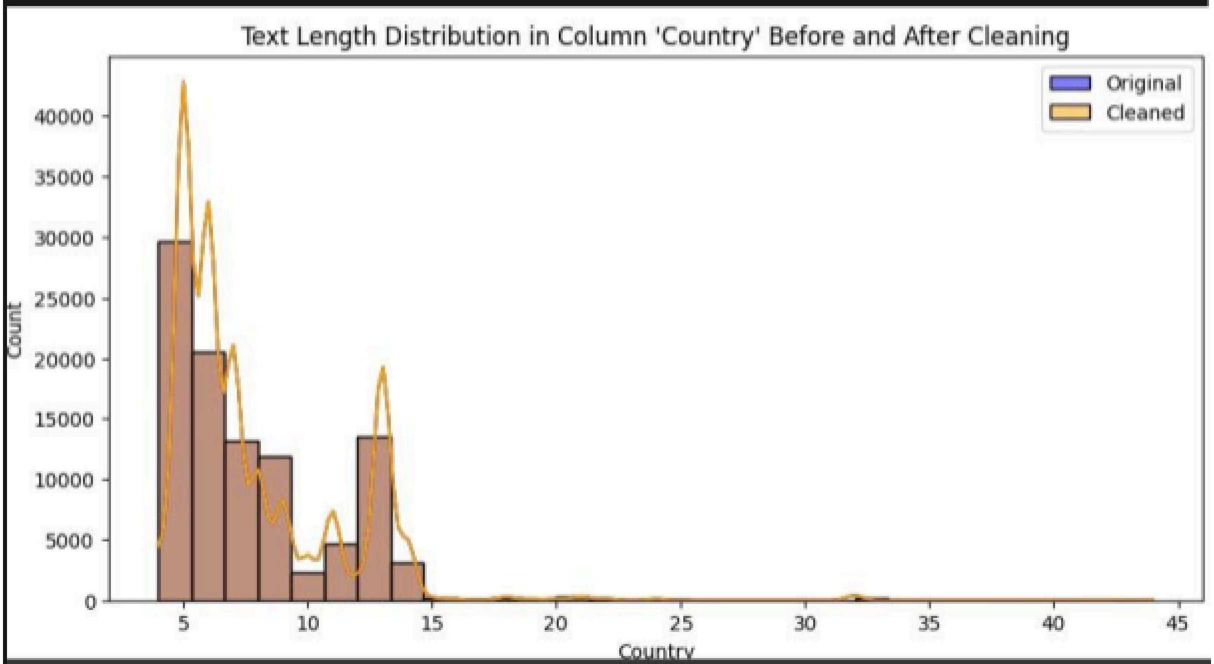

```
# 3. Text Cleaning
def clean_text(text):
    text = text.lower()  # Convert to lowercase
    text = re.sub(r'http\S+|www\S+', '', text)  # Remove
    URLs text = re.sub(r'[^a-zA-Z\s]', '', text)# Remove
    special
characters and numbers
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra
    spaces return text

text_columns = [col for col in data.columns if data[col].dtype
== 'object']  # Identify text columns
for col in text_columns:
    data[col] = data[col].apply(clean_text)
    print(f"Cleaned text in column '{col}'")
# Show updated dataset after text cleaning
print("Dataset after text cleaning:")
print(data.head())
for col in text_columns:
    original_text_lengths = original_data[col].dropna().apply(lambda x:
len(str(x)))
    cleaned_text_lengths = data[col].apply(lambda x: len(str(x)))
    plt.figure(figsize=(10, 5))
    sns.histplot(original_text_lengths, color='blue', label='Original',
kde=True, bins=30)
    sns.histplot(cleaned_text_lengths, color='orange', label='Cleaned',
kde=True, bins=30)
```

```
    plt.title(f"Text Length Distribution in Column '{col}' Before and
After Cleaning")
    plt.legend()
    plt.show()
```



Text Length Distribution in Column 'City' Before and After Cleaning





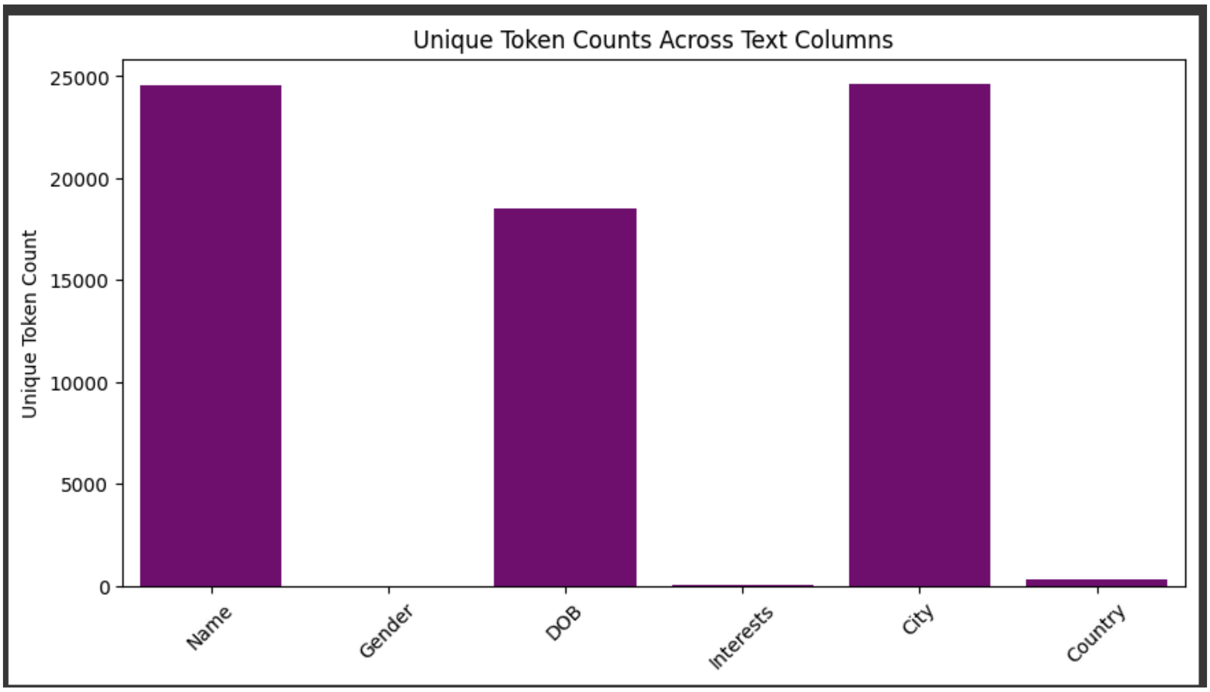

```
# 4. Tokenization and Lemmatization
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))

def tokenize_and_lemmatize(text):
    tokens = word_tokenize(text)
    tokens = [word for word in tokens if word not in stop_words]
# Remove stopwords
```

```
    tokens = [lemmatizer.lemmatize(word) for word in tokens]
    #
Lemmatize tokens
    return tokens

# Ensure 'Name' is included in text_columns if it's intended to be
tokenized
text_columns = [col for col in data.columns if data[col].dtype ==
'object']
if 'Name' not in text_columns and 'Name' in data.columns:
    text_columns.append('Name')

for col in text_columns:
    data[col + '_tokens'] = data[col].apply(tokenize_and_lemmatize)
    print(f"Tokenized and lemmatized text in column '{col}'")
# Show updated dataset after tokenization and lemmatization
print("Dataset after tokenization and lemmatization:")
print(data.head())
```

```
Tokenized and lemmatized text in column 'Country'
Dataset after tokenization and lemmatization:
   UserID               Name  Gender         DOB  \
0       1      Jesse Lawhorn  Female  1958-10-15
1       2        Stacy Payne  Female  2004-07-21
2       3  Katrina Nicewander  Female  2000-02-07
3       4      Eric Yarbrough    Male  1985-04-14
4       5      Daniel Adkins  Female  1955-09-18

                                         Interests         City    Country  \
0            'Movies', 'Fashion', 'Fashion', 'Books'      Sibolga  Indonesia
1  'Gaming', 'Finance and investments', 'Outdoor ...     Al Abyār      Libya
2    'DIY and crafts', 'Music', 'Science', 'Fashion'  Wādī as Sīr     Jordan
3       'Outdoor activities', 'Cars and automobiles'       Matera      Italy
4                             'Politics', 'History'      Biruaca  Venezuela

            Name_tokens Gender_tokens    DOB_tokens  \
0      [Jesse, Lawhorn]      [Female]  [1958-10-15]
1        [Stacy, Payne]      [Female]  [2004-07-21]
2  [Katrina, Nicewander]     [Female]  [2000-02-07]
3     [Eric, Yarbrough]        [Male]  [1985-04-14]
4      [Daniel, Adkins]      [Female]  [1955-09-18]

                                   Interests_tokens  City_tokens  \
0  ['Movies, ', ,, 'Fashion, ', ,, 'Fashion, ', ,...    [Sibolga]
1  ['Gaming, ', ,, 'Finance, investment, ', ,, 'O...  [Al, Abyār]
2  ['DIY, craft, ', ,, 'Music, ', ,, 'Science, ',...  [Wādī, Sīr]
3   ['Outdoor, activity, ', ,, 'Cars, automobile, ']     [Matera]
4                     ['Politics, ', ,, 'History, ']    [Biruaca]

  Country_tokens
0    [Indonesia]
```

unique_tokens_counts = {col: len(set(token for tokens in data[col + '_tokens'] for token in tokens)) for col in text_columns}

```
plt.figure(figsize=(10, 5))
sns.barplot(x=list(unique_tokens_counts.keys()),
y=list(unique_tokens_counts.values()), color='purple')
plt.title("Unique Token Counts Across Text Columns")
plt.ylabel("Unique Token Count")
plt.xticks(rotation=45)
plt.show()
```



Conclusion:

This experiment we successfully demonstrated the importance of data cleaning in handling social media data using Python. By addressing missing values, removing duplicates, standardizing formats, and preprocessing text, we improved data quality and usability. Tokenization and lemmatization helped refine the text for further analysis. Ensuring data accuracy, consistency, and reliability enhances the value of insights derived from social media datasets. This process highlights the crucial role of data preprocessing in enabling meaningful data-driven decision-making.