Name:- Arpita Nilesh Tap
PRN:- 9024010700004
Batch- ET13
Roll No- 47

Page No.
Date 28/04/2025

Topic:- The Blog Authorship Corpus

1) load the Blog Authorship corpus file.

```
import Pandas as Pd.
#Assuming the dataset is in csv format
data = Pd.read_csv('blog_authorship_corpus.c sv')
print(data.head())
```

2) Display total no of blogs

```
print(f"Total number of blogs: {len(data)}")
```

3) Find all unique authors

```
unique_authors = data['author'].unique()
print(f"Number of unique authors: {len(unique_authors)}")
```

4) Find number of blogs written by males

```
male_blogs = data[data['gender'] == 'male']
print(f"number of blogs by males: {len(male_blogs)}")
```

5) Find average age of authors

```
average_age = data['age'].mean()
print(f"Average age of authors: {average_age:.0f}")
```

Shrikrupa

Page No.
Date / /

6) Find the most common industry among authors
→
```
common_industry = data['industry'].mode()[0]
print(f"Most common industry: {common_industry}")
```

7) create a bar plot showing gender distribution
→
```
import matplotlib.pyplot as plt
data['gender'].value_counts().plot(kind='bar')
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Name of Blogs')
plt.show()
```

8) create a new column: blog length in words
→
```
data['word_count'] =
data['clean_text'].apply(lambda x: len(x.split()))
print(data[['word_count']].describe())
```

9) Find the author with maximum blog posts
→
```
top_author = data['author'].value_counts().idxmax()
print(f"Author with most blogs: {top_author}")
```

10) Find average blog length by gender
→
```
avg_length_by_gender = data.groupby('gender')
['word_count'].mean()
print(avg_length_by_gender)
```

Shrikrupa

Page No.
Date / /

11) Find correlation b/w age and blog length

```
correlation = data['age'].corr(data['word_count'])
print(f"correlation between age and word count: {correlation:.2f}")
```

12) Save cleaned data to csv

```
data.to_csv('cleaned_blog_authorship.csv', index=False)
print("cleaned data saved to 'cleaned_blog_authorship.csv'")
```

13) Calculate percentage of blogs written by authors under 25

```
under_25 = data[data['age'] < 25]
percentage = (len(under_25) / len(data)) * 100
print(f"percentage of blogs by authors under 25: {percentage:.2f}%.")
```

14) Clean text: remove special characters and nos

```
def clean_text(text):
    return re.sub(r'[^a-zA-Z\s]', '', text)
data['clean_text'] = data['text'].apply(clean_text)
print(data[['text', 'clean_text']].head())
```

Page No.
Date / /

15) Plot histogram of blog lengths

```
data['word_count'].plot(kind='hist', bins=30)
plt.title('Histogram of Blog Lengths')
plt.xlabel('word count')
plt.show()
```

16) Find longest blog post

```
longest_blog = data.loc[data['word_count'].
idxmax()]
print("Longest Blog Author:", longest_blog['author'])
print("word count:", longest_blog['word_count'])
```

17) Detect language blogs (assuming non-english blogs)

```
from langdetect import detect
data['language'] = data['text'].apply(lambda x:
detect(x))
print(data['language'].value_counts())
```

18) Find blogs that are very short (less than 50 word

```
short_blogs = data[data['word_count'] < 50]
print(short_blogs[['author', 'word_count']].head(
```

Page No.
Date / /

19) Visualize blog counts by age group

→
```
sns.countplot( x='age_group', data=data, palette='set2')
plt.title('Blog counts by Age Group')
plt.show()
```

---

20) compute readability score (Flesch Reading ease)

→
```
import textstat
data['readability'] = data['clean_text'].apply(textstat.
flesch_reading_ease)
print(data[['readability']].describe())
```
