EDS Theory Assignment

---

**Blog Authorship Corpus Data Analysis**

---

Name:Atharva Gajbhiye

Divison:CS3

Roll_no: CS3-42

PRN: 202401040259

Loading the dataset.

```
#mount the google datset
from google.colab import drive
drive.mount('/content/drive')

# load the dataset
!pip install numpy pandas
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

path='/content/drive/MyDrive/blogtext.csv'
df=pd.read_csv(path)

df.info()
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (2.0.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 681284 entries, 0 to 681283
Data columns (total 7 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   id      681284 non-null  int64
 1   gender  681284 non-null  object
 2   age     681284 non-null  int64
 3   topic   681284 non-null  object
 4   sign    681284 non-null  object
 5   date    681284 non-null  object
 6   text    681284 non-null  object
dtypes: int64(2), object(5)
memory usage: 36.4+ MB
```

Problem statements

1. Display First 15 entries and and show variance of age distribution across industries.

```
#1. Display first 15 entries and show variance of age distribution across industries.
print("Problem 1: Display first 15 entries and show variance of age distribution across industries.")
df.head(15)
variance=df.groupby('topic')['age'].var().round(2)
print(variance)
```

```
Problem 1: Display first 15 entries and show variance of age distribution across industries.
topic
Accounting                 108.83
Advertising                 55.32
Agriculture                 41.81
Architecture                36.43
Arts                        47.94
Automotive                  59.21
Banking                     17.43
Biotech                     44.71
```

```
BusinessServices            64.36
Chemicals                   21.13
Communications-Media        45.34
Construction                65.73
Consulting                  90.75
Education                   64.55
Engineering                 20.30
Environment                 27.98
Fashion                     63.66
Government                  55.00
HumanResources              68.15
Internet                    65.04
InvestmentBanking           37.83
Law                         24.04
LawEnforcement-Security     72.85
Manufacturing               35.69
Maritime                    60.52
Marketing                   30.80
Military                    83.15
Museums-Libraries           77.65
Non-Profit                  54.95
Publishing                  51.46
RealEstate                  56.76
Religion                    71.72
Science                     22.40
Sports-Recreation           45.12
Student                     16.66
Technology                  45.92
Telecommunications          41.67
Tourism                     48.90
Transportation              47.31
indUnk                      63.86
Name: age, dtype: float64
```

2. Display data type of each column.

```
#2. Display data type of each column.
print("Problem 2: Display data type of each column")
df.dtypes
```

```
Problem 2: Display data type of each column
```

| | 0 |
|---|---|
| **id** | int64 |
| **gender** | object |
| **age** | int64 |
| **topic** | object |
| **sign** | object |
| **date** | object |
| **text** | object |

**dtype:** object

3. Checking for missing values in all columns.

```
#Checking for missing values in all columns.
print("Problem 3: Checking for missing values in all columns.")
df.isnull().sum()
```

```
Problem 3: Checking for missing values in all columns.
```

| | 0 |
|---|---|
| id | 0 |
| gender | 0 |
| age | 0 |
| topic | 0 |
| sign | 0 |
| date | 0 |
| text | 0 |

dtype: int64

4. Find the unique bloggers by gender and and list them all.

```
# Find the unique bloggers by gender and and list them all.
print("Problem 4: Find the unique bloggers by gender and and list them all.")
print(df['gender'].unique())
df['gender'].value_counts()
```

```
Problem 4: Find the unique bloggers by gender and and list them all.
['male' 'female']
```

| gender | count |
|---|---|
| male | 345193 |
| female | 336091 |

dtype: int64

Double-click (or enter) to edit

5. Find the minimun and maximum bloggers age.

```
#Find the minimun and maximum  bloggers age.
print("Problem 5: Find the minimun and maximum  bloggers age.")
print(df['age'].head(5))
print("Minimum age:",df['age'].min())
print("Maximum age:",df['age'].max())
```

```
Problem 5: Find the minimun and maximum  bloggers age.
0    15
1    15
2    15
3    15
4    33
Name: age, dtype: int64
Minimum age: 13
Maximum age: 48
```

6.Calculate the average , median, and mode of the blooger's age.

```
#Calculate the average , median, and mode of the blooger's age.
print("Problem 6: Calculate the average , median, and mode of the blooger's age.")
print("Average age:",np.mean(df['age']).round(2))
print("Median age:",np.median(df['age']))
print("Mode age:",df['age'].mode()[0])
```

```
Problem 6: Calculate the average , median, and mode of the blooger's age.
Average age: 23.93
Median age: 24.0
Mode age: 17
```

7. Gender distribution in percentage.

```
#Gender distribution in percentage.
print("Problem 7: Gender distribution (percentages)\n")
gender_dist = df['gender'].value_counts(normalize=True) * 100
display(gender_dist)
```

```
Problem 7: Gender distribution (percentages)
```

| gender | proportion |
|---|---|
| male | 50.668003 |
| female | 49.331997 |

dtype: float64

8. Top 10 industries

```
#Top 10 industries
print("Problem 8: Top 10 industries by blogger count\n")
print(df['topic'].value_counts().head(10))
```

```
Problem 8: Top 10 industries by blogger count

topic
indUnk                  251015
Student                 153903
Technology               42055
Arts                     32449
Education                29633
Communications-Media     20140
Internet                 16006
Non-Profit               14700
Engineering              11653
Law                       9040
Name: count, dtype: int64
```

9. Drop duplicate blogger IDs

```
#Drop duplicate blogger IDs
print("Problem 9: Drop duplicate IDs\n")
df = df.drop_duplicates(subset='id')
print("Remaining records after duplicate removal:",len(df))
print(df[['id','gender','age','topic','sign','date']].head(15))
```

```
Problem 9: Drop duplicate IDs

Remaining records after duplicate removal: 19320
          id  gender  age              topic         sign            date
0    2059027    male   15            Student          Leo     14,May,2004
4    3581210    male   33  InvestmentBanking     Aquarius    11,June,2004
74   3539003  female   14             indUnk        Aries    07,June,2004
95   4172416  female   25             indUnk    Capricorn  08,August,2004
97   3668238  female   17            Student       Gemini    30,June,2004
107  4030905  female   17            Student        Aries    31,July,2004
132  4198080  female   23             indUnk     Aquarius  10,August,2004
133  3705830    male   25         Non-Profit       Cancer    29,June,2004
179  3649763  female   33            Banking     Aquarius    20,June,2004
192  3389918  female   37             indUnk     Aquarius     23,May,2004
211  4304458  female   25             indUnk  Sagittarius  23,August,2004
214  3429420    male   15            Student     Aquarius     30,May,2004
248  3389671    male   26             indUnk          Leo     28,May,2004
283   649790  female   24             indUnk      Scorpio     31,May,2002
474  3022585  female   27          Education     Aquarius  01,August,2004
```

10. Create 'decade' column based on age

```
#Create 'decade' column based on age
print("Problem 10: Creating 'decade' column\n")
df.loc[:,'decade'] = (df['age'] // 10) * 10
print(df[['age', 'decade']].head(15))
```

```
Problem 10: Creating 'decade' column

     age  decade
0     15      10
4     33      30
74    14      10
95    25      20
97    17      10
107   17      10
132   23      20
133   25      20
179   33      30
192   37      30
211   25      20
214   15      10
248   26      20
283   24      20
474   27      20
```

11. Number of bloggers with posts exceeding 5000 words

```
#Number of bloggers with posts exceeding 5000 words
print("Problem 11: Bloggers with posts >5000 words\n")
df.loc[:,'word_count'] = df['text'].apply(lambda x: len(str(x).split()))
print(f"Bloggers with >5000 words: {(df['word_count'] > 5000).sum()}")

print(df['word_count'])
```

```
Problem 11: Bloggers with posts >5000 words

Bloggers with >5000 words: 20
0          28
4          65
74        185
95         16
97         41
          ...
681004    257
681025    874
681141      7
681156    415
681161     13
Name: word_count, Length: 19320, dtype: int64
```

12.Create 'text_length' column

```
#Create 'text_length' column
print("Problem 12: Create 'text_length' (characters)\n")
df.loc[:,'text_length'] = df['text'].apply(lambda x: len(str(x)))
display(df[['id', 'text_length']].head())
```

```
Problem 12: Create 'text_length' (characters)
```

| | id | text_length |
|---|---|---|
| **0** | 2059027 | 157 |
| **4** | 3581210 | 402 |
| **74** | 3539003 | 932 |
| **95** | 4172416 | 103 |
| **97** | 3668238 | 476 |

13. Calculate average number of sentences per blog post

```
#Calculate average number of sentences per blog post
print("Problem 13: Average sentences per post\n")
df.loc[:,'sentence_count'] = df['text'].apply(lambda x: str(x).count('.')+str(x).count('!')+str(x).count('?'))
print("Average sentences per blog post:" ,df['sentence_count'].mean().round(2))
```

```
Problem 13: Average sentences per post

Average sentences per blog post: 33.63
```

14. Average text length by gender

```
#Average text length by gender
print("Problem 9: Average text length by gender")
print(df.groupby('gender')['text_length'].mean().round(2))
```

```
Problem 9: Average text length by gender
gender
female    1596.30
male      1612.39
Name: text_length, dtype: float64
```

15. Most common industry by decade

```
#Most common industry by decade
print("Problem 15: Most common industry per age decade\n")
common = df.groupby('decade')['topic'].agg(lambda x: x.mode()[0])
display(common)
```

```
Problem 15: Most common industry per age decade
```

| decade | topic |
|---|---|
| 10 | Student |
| 20 | indUnk |
| 30 | indUnk |
| 40 | indUnk |

dtype: object

16.Create a pivot table: average age and average word count per gender and industry

```
#Create a pivot table: average age and average word count per gender and industry
print("Problem 16: Pivot table of age and word count\n")
pivot = pd.pivot_table(df.head(30), index=['gender', 'topic'], values=['age', 'word_count'], aggfunc='mean')
print(pivot.round(2))
```

```
Problem 16: Pivot table of age and word count

                                  age  word_count
gender topic
female Banking               33.00      196.00
       Education             27.00      550.00
       Student               16.33      248.67
       indUnk                26.00      472.92
male   BusinessServices      24.00      386.00
       Communications-Media  41.00      723.00
       Engineering           24.00       30.00
       InvestmentBanking     33.00       65.00
       Non-Profit            25.00      270.00
       Science               15.00       47.00
       Sports-Recreation     17.00       95.00
       Student               15.25      242.75
       indUnk                24.50      246.00
```

17. Blogger with the maximum number of posts

```
#Blogger with the maximum number of posts
print("Problem 17: Blogger with most posts\n")
topposts = df['id'].value_counts().idxmax()
print("Blogger with most posts:",topposts)
```

```
Problem 17: Blogger with most posts

Blogger with most posts: 1713845
```

18. Percentage of short posts greater than 200 words

```
# Percentage of short posts greater than 200 words
print("Problem 18: Percentage of short posts (<200 words)\n")
short_posts = (df['word_count'] < 200).mean() * 100
print("Percentage:", short_posts.round(2))
```

```
Problem 18: Percentage of short posts (<200 words)

Percentage: 54.47
```

19. How many blog posts mention 'life'.

```
#How many blog posts mention 'life'
print("Problem 19: Posts containing 'life'\n")
life_posts = df['text'].str.lower().str.contains('life')
print("Number of posts mentioning 'life':", life_posts.sum())
```

```
Problem 19: Posts containing 'life'

Number of posts mentioning 'life': 4457
```

20. Create a new column of name Word density (words per character).

```
#Create a new column of name  Word density (words per character).
print("Problem 20: Word density (words per character)")
df.loc[:,'word_density'] = df['word_count'] / df['text_length']
print(df[['word_density']].head(15))
```

```
Problem 20: Word density (words per character)
     word_density
0        0.178344
4        0.161692
74       0.198498
95       0.155340
97       0.086134
107      0.164539
132      0.193997
133      0.194665
179      0.177215
192      0.187382
211      0.173625
214      0.181200
248      0.184008
283      0.147826
474      0.187521
```

**Thank You**

+ Code + Text