EDS THEORY ASSIGNMENT

Name: Rutuja Balasaheb Dukare

Roll No: ET2-80

PRN: 202401070188

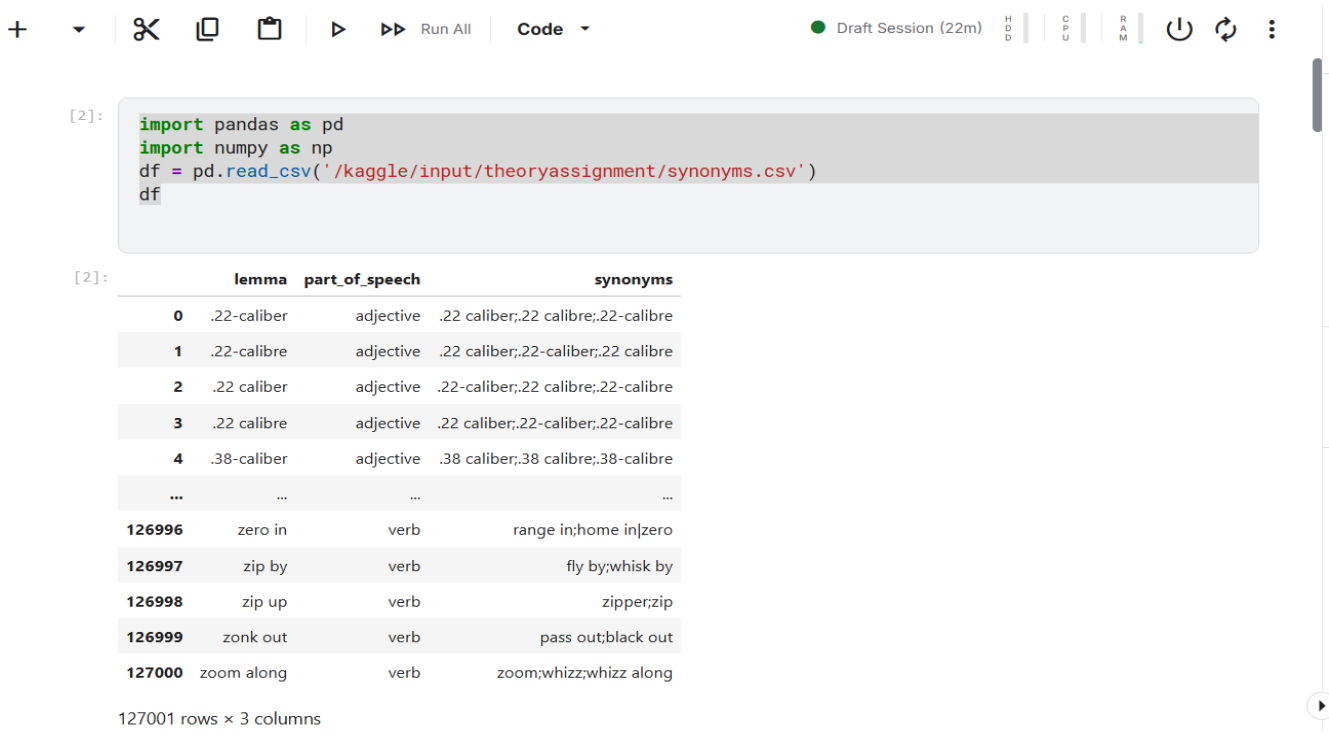
Dataset link:
https://drive.google.com/drive/folders/1qlN1Vs_l5Bun2x9bp24mltE8PXP0GxgG

import pandas as pd

import numpy as np

df = pd.read_csv('/kaggle/input/theoryassignment/synonyms.csv')

df



	lemma	part_of_speech	synonyms
0	.22-caliber	adjective	.22 caliber;.22 calibre;.22-calibre
1	.22-calibre	adjective	.22 caliber;.22-caliber;.22 calibre
2	.22 caliber	adjective	.22-caliber;.22 calibre;.22-calibre
3	.22 calibre	adjective	.22 caliber;.22-caliber;.22-calibre
4	.38-caliber	adjective	.38 caliber;.38 calibre;.38-calibre
...	...	...	...
126996	zero in	verb	range in;home in|zero
126997	zip by	verb	fly by;whisk by
126998	zip up	verb	zipper;zip
126999	zonk out	verb	pass out;black out
127000	zoom along	verb	zoom;whizz;whizz along

127001 rows × 3 columns

1. Find the total number of missing values in each column.
   missing_values = df.isnull().sum()
   missing_values

```
missing_values = df.isnull().sum()
missing_values
```

```
[28]: lemma            3
      part_of_speech    0
      synonyms          2
      dtype: int64
```

2. Replace the missing values in ‘lemma’ and ‘synonyms’ with “unknown”.

   df_filled = df.fillna('unknown')
   df_filled.isnull().sum()

```
[5]: df_filled = df.fillna('unknown')
     df_filled.isnull().sum()
```

```
[5]: lemma            0
     part_of_speech    0
     synonyms          0
     dtype: int64
```

3. Find the number of unique parts of speech.
   unique_pos_count = df_filled['part_of_speech'].nunique()
   unique_pos_count

```
unique_pos_count = df_filled['part_of_speech'].nunique()
unique_pos_count
```

```
[6]: 5
```

4. Find the top 5 most common parts of speech.
   top_5_pos = df_filled['part_of_speech'].value_counts().head(5)
   top_5_pos

```
top_5_pos = df_filled['part_of_speech'].value_counts().head(5)
top_5_pos
```

```
[29]: part_of_speech
      noun         94600
      verb         13681
      satellite    11752
      adjective     4274
      adverb        2694
      Name: count, dtype: int64
```

5. Find how many lemmas have more than 3 synonyms.
   df_filled['synonym_list'] = df_filled['synonyms'].apply(lambda x: x.split(';'))

lemmas_with_more_than_3_synonyms = (df_filled['synonym_list'].apply(len) > 3).sum()
lemmas_with_more_than_3_synonyms

```
df_filled['synonym_list'] = df_filled['synonyms'].apply(lambda x: x.split(';'))
lemmas_with_more_than_3_synonyms = (df_filled['synonym_list'].apply(len) > 3).sum()
lemmas_with_more_than_3_synonyms
```

[8]: 29181

6. Create a new column showing the number of synonyms for each lemma.
   df_filled['num_synonyms'] = df_filled['synonym_list'].apply(len)
   df_filled[['lemma', 'num_synonyms']].head()

```
df_filled['num_synonyms'] = df_filled['synonym_list'].apply(len)
df_filled[['lemma', 'num_synonyms']].head()
```

[10]:

	lemma	num_synonyms
0	.22-caliber	3
1	.22-calibre	3
2	.22 caliber	3
3	.22 calibre	3
4	.38-caliber	3

7. Find the average number of synonyms per lemma.
   avg_synonyms_per_lemma = df_filled['num_synonyms'].mean()
   avg_synonyms_per_lemma

```
avg_synonyms_per_lemma = df_filled['num_synonyms'].mean()
avg_synonyms_per_lemma
```

[11]: 2.9391973291548887

8. Use lemmas whose synonyms include the word “caliber”.
   lemmas_with_caliber = df_filled[df_filled['synonyms'].str.contains('caliber', case=False)]['lemma'].tolist()
   lemmas_with_caliber[:5]

```
lemmas_with_caliber = df_filled[df_filled['synonyms'].str.contains('caliber', case=False)]['lemma'].toli
lemmas_with_caliber[:5]  # First 5
```

[12]: ['.22-caliber', '.22-calibre', '.22 caliber', '.22 calibre', '.38-caliber']

9. Find the longest lemma by character length.
   longest_lemma = df_filled['lemma'].iloc[df_filled['lemma'].str.len().idxmax()]
   longest_lemma

```
longest_lemma = df_filled['lemma'].iloc[df_filled['lemma'].str.len().idxmax()]
longest_lemma
```

```
[14]: 'blood-oxygenation level dependent functional magnetic resonance imaging'
```

10. Find the shortest synonym list by character length.
    shortest_synonym_list =
    df_filled['synonyms'].iloc[df_filled['synonyms'].str.len().idxmin()]
    shortest_synonym_list

```
shortest_synonym_list = df_filled['synonyms'].iloc[df_filled['synonyms'].str.len().idxmin()]
shortest_synonym_list
```

```
[15]: '0'
```

11. Group by part_of_speech and find the average synonyms per lemma.
    avg_synonyms_by_pos =
    df_filled.groupby('part_of_speech')['num_synonyms'].mean()
    avg_synonyms_by_pos

```
[16]: avg_synonyms_by_pos = df_filled.groupby('part_of_speech')['num_synonyms'].mean()
      avg_synonyms_by_pos
```

```
[16]: part_of_speech
      adjective    1.453205
      adverb       2.220119
      noun         2.589989
      satellite    3.234683
      verb         5.705869
      Name: num_synonyms, dtype: float64
```

12. Find the most common synonym across the entries.
    all_synonyms_flat = [syn for sublist in df_filled['synonym_list'] for syn in sublist]
    most_common_synonym = pd.Series(all_synonyms_flat).value_counts().idxmax()
    most_common_synonym

```
all_synonyms_flat = [syn for sublist in df_filled['synonym_list'] for syn in sublist]
most_common_synonym = pd.Series(all_synonyms_flat).value_counts().idxmax()
most_common_synonym
```

```
']: 'pass'
```

13. Find lemmas where all synonyms are identical.
    identical_synonyms = df_filled[df_filled['synonym_list'].apply(lambda x: len(set(x)) == 1)]['lemma'].tolist()
    identical_synonyms[:5]

```
identical_synonyms = df_filled[df_filled['synonym_list'].apply(lambda x: len(set(x)) == 1)]['lemma'].tol
identical_synonyms[:5]  # First 5
```

```
[18]: ['0', '10-membered', '1000th', '101st', '105th']
```

14. Extract all lemmas whose first character is a digit.
    lemmas_starting_with_digit =
    df_filled[df_filled['lemma'].str[0].str.isdigit()]['lemma'].tolist()
    lemmas_starting_with_digit[:5]

```
lemmas_starting_with_digit = df_filled[df_filled['lemma'].str[0].str.isdigit()]['lemma'].tolist()
lemmas_starting_with_digit[:5]  # First 5
```

```
[19]: ['0', '0', '1', '1', '10']
```

15. Create a new column with the first synonym only.
    df_filled['first_synonym'] = df_filled['synonym_list'].apply(lambda x: x[0])
    df_filled[['lemma', 'first_synonym']].head()

```
df_filled['first_synonym'] = df_filled['synonym_list'].apply(lambda x: x[0])
df_filled[['lemma', 'first_synonym']].head()
```

[20]:

	lemma	first_synonym
0	.22-caliber	.22 caliber
1	.22-calibre	.22 caliber
2	.22 caliber	.22-caliber
3	.22 calibre	.22 caliber
4	.38-caliber	.38 caliber

16. Replace all semicolon (;) separators in synonyms with commas .
    df_filled['synonyms_commas'] = df_filled['synonyms'].str.replace(';', ',', regex=False)
    df_filled[['lemma', 'synonyms_commas']].head()

```
df_filled['synonyms_commas'] = df_filled['synonyms'].str.replace(';', ',', regex=False)
df_filled[['lemma', 'synonyms_commas']].head()
```

[21]:

	lemma	synonyms_commas
0	.22-caliber	.22 caliber,.22 calibre,.22-calibre
1	.22-calibre	.22 caliber,.22-caliber,.22 calibre
2	.22 caliber	.22-caliber,.22 calibre,.22-calibre
3	.22 calibre	.22 caliber,.22-caliber,.22-calibre
4	.38-caliber	.38 caliber,.38 calibre,.38-calibre

17. Count how many lemmas contain hyphens.
    lemmas_with_hyphen_count = df_filled['lemma'].str.contains('-').sum()
    lemmas_with_hyphen_count

```
lemmas_with_hyphen_count = df_filled['lemma'].str.contains('-').sum()
lemmas_with_hyphen_count
```

[23]: 5243

18. Find how many lemmas are completely lowercase.
    lowercase_lemmas_count = df_filled['lemma'].apply(lambda x: x.islower()).sum()
    lowercase_lemmas_count

```
lowercase_lemmas_count = df_filled['lemma'].apply(lambda x: x.islower()).sum()
lowercase_lemmas_count
```

[24]: 126824

19. Find the lemmas with the maximum number of characters among synonyms.
    lemma_max_synonym_chars =
    df_filled.iloc[df_filled['synonyms'].str.len().idxmax()]['lemma']
    lemma_max_synonym_chars

```
lemma_max_synonym_chars = df_filled.iloc[df_filled['synonyms'].str.len().idxmax()]['lemma']
lemma_max_synonym_chars
```

[26]: 'broke'

20. Using NumPy , create an array of synonym counts and compute its mean and standard deviation.
    import numpy as np

    synonym_counts_array = np.array(df_filled['num_synonyms'])
    synonym_counts_mean = np.mean(synonym_counts_array)
    synonym_counts_std = np.std(synonym_counts_array)

    synonym_counts_mean, synonym_counts_std

```
import numpy as np

synonym_counts_array = np.array(df_filled['num_synonyms'])
synonym_counts_mean = np.mean(synonym_counts_array)
synonym_counts_std = np.std(synonym_counts_array)

synonym_counts_mean, synonym_counts_std
```

```
27]: (2.9391973291548887, 3.1703255388271194)
```