In [41]:

```
import numpy as np
import pandas as pd
import requests
from bs4 import BeautifulSoup
import re
```

In [42]:

```
urls=pd.read_excel(r"E:\Black_Coffer_Assignment\Input.xlsx")
urls
```

Out[42]:

	URL_ID	URL
0	37	https://insights.blackcoffer.com/ai-in-healthc...
1	38	https://insights.blackcoffer.com/what-if-the-c...
2	39	https://insights.blackcoffer.com/what-jobs-wil...
3	40	https://insights.blackcoffer.com/will-machine-...
4	41	https://insights.blackcoffer.com/will-ai-repla...
...	...	...
109	146	https://insights.blackcoffer.com/blockchain-fo...
110	147	https://insights.blackcoffer.com/the-future-of...
111	148	https://insights.blackcoffer.com/big-data-anal...
112	149	https://insights.blackcoffer.com/business-anal...
113	150	https://insights.blackcoffer.com/challenges-an...

114 rows × 2 columns

In [43]:

```
path=r"E:/Black_Coffer_Assignment/Web_Scrapped_Articles/"
for i in range(0,len(urls)):
    position=i
    row=urls.iloc[position]
    url_id=row[0]
    url=row[1]
    agent = {"User-Agent": 'Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safar
    page=requests.get(url, headers=agent)
    soup=BeautifulSoup(page.content,'html.parser')
    if len(list(soup.findAll(attrs = {"class":"td-post-content"})))!=0:
        Title=BeautifulSoup(page.content,'html.parser').h1.text
        Article=soup.findAll(attrs = {"class":"td-post-content"})[0].text.replace("\n"," ")
        content=[Title, Article]
        f = open(path+''+str(url_id)+".txt", 'w')
        f.write(' '.join(content))
```

Stop_Words

In [44]:

```
import os
path1=r'E:/Black_Coffer_Assignment/StopWords'
all_files = os.listdir(path1)

new_list=[]
for fle in all_files:
    with open(os.path.join(path1, fle),"rb") as f:
        text = f.read()
        new_list.append(text)

stopwords=str(new_list).replace("\\n"," ").replace("\\r"," ").replace("|","").replace(",",
            '').replace("b'",'').replace("'",'').replace("[",'').replace("]",'').strip().lower().split()
```

In [45]:

```
len(stopwords)
```

Out[45]:

```
14238
```

In [ ]:

```

```

In [46]:

```
import os
path3=r'E:/Black_Coffer_Assignment/MasterDictionary/'
all_files2 = os.listdir(path3)

negative=open(path3+all_files2[0],'rb').read()
negative_words=str(negative).replace("\\n"," ").replace("\\r"," ").replace("|","").replace(",",
            '').replace("b'",'').replace("'",'').replace("[",'').replace("]",'').strip().lower().split()

positive=open(path3+all_files2[1],'rb').read()
positive_words=str(positive).replace("\\n"," ").replace("\\r"," ").replace("|","").replace(",",
            '').replace("b'",'').replace("'",'').replace("[",'').replace("]",'').strip().lower().split()
```

Positive Score, Negative Score, Polarity Score, Subjectivity Score

In [47]:

```
word_dict = {'Positive':[], 'Negative':[]}
def add_values_in_dict(word_dict, key, list_of_words):
    if key not in word_dict:
        word_dict[key] = list()
    word_dict[key].append(list_of_words)
    return word_dict
```

In [48]:

```
def count_syllables(word):
    c = 0
    vowels = 'aeiou'
    l = re.findall(f'(?!e$)(?!es$)(?!ed$)[{vowels}]', word, re.I)
    return len(l)
```

In [49]:

```
import os
import json
path2=r'E:/Black_Coffer_Assignment/Web_Scrapped_Articles/'
all_files1 = os.listdir(path2)

positive_score=[]
negative_score=[]
polarity_score=[]
subjectivity_score=[]
url_ids=[]
syllable_count=[]

for file in all_files1:
    url_ids.append(int(file.replace(".txt",'')))
    with open(os.path.join(path2, file),"rb") as p:
        text1 = p.read()
        text2=str(text1).lower().replace("/",
                        ' ').replace("b'",' ').replace("\\",' ').replace(".",
                        ' ').replace(":"," ").replace("%",' ').replace("-",
                        ' ').replace(",",' ').replace("$"," ").replace("[",' ').replace("]",
                        ' ').replace("(",' ').replace(")",' ').replace("!",' ').replace("\'",
                        ' ').replace("&",' ').replace("xe2"," ").replace("x80",' ').replace("x93",' ').replace("x99",
                        ' ').replace("x98",' ').replace("xc2 xa0 xc2 xa0",' ').replace("xc2 xa0",' ').replace("x9c",
                        ' ').replace("x9d",' ').replace("?",' ').replace("x94",' ').replace("fy20",
                        ' ').replace("fy21",' ').replace("fy22",' ').replace("xa6",' ').replace("*",' ').strip().split()
        pos=0
        neg=0
        pol=0
        cleaned_word=0
        for word in text2:
            if word not in stopwords:
                cleaned_word=cleaned_word+1
                list1 = []
                list1.append(count_syllables(word))
                if word in positive_words:
                    word_dict = add_values_in_dict(word_dict, 'Positive', word)
                    pos=pos+1
                elif word in negative_words:
                    word_dict = add_values_in_dict(word_dict, 'Negative', word)
                    neg=neg-1
        pol = (pos - (neg*-1))/ ((pos + (neg*-1)) + 0.000001)
        sub = (pos + (neg*-1))/ (cleaned_word + 0.000001)

    file1 = open(r'E:\Black_Coffer_Assignment\dictionary.json', 'w')  #Creating Dictionary of Positive and Negative Words
    json.dump(word_dict,file1)
    file1.close()

    positive_score.append(pos)      #Positive Score
    negative_score.append(neg*-1)   #Negative Score
    polarity_score.append(pol)      #Polarity Score
    subjectivity_score.append(sub)  #Subjectivity Score
    syllable_count.append(sum(list1)/len(list1))  #Syllable count
```

In [76]:

```
df1=pd.DataFrame({'URL_ID':url_ids,'POSITIVE SCORE':positive_score,
                 'NEGATIVE SCORE':negative_score, 'POLARITY SCORE':polarity_score,
                'SUBJECTIVITY SCORE':subjectivity_score})

df3=pd.DataFrame({'URL_ID':url_ids, 'SYLLABLE PER WORD':syllable_count})
```

In [ ]:

```
```

Average Sentence Length, Percentage of Complex Words, Fog Index, Complex Word Count, Word Count, Personal Pronoun, Average Word Lenth

In [50]:

```
def count_complex_words(words_list):
    c = 0
    for word in words_list:
        l = re.findall('(?!e$)[aeiou]+', word, re.I)+re.findall('^[aeiouy]*e$', word, re.I)
        if len(l) > 2:
            c += 1
    return c
```

In [51]:

```
def count_personal_pronouns(text):
    pronoun_count = re.compile(r'\b(I|we|ours|my|mine|(?-i:us))\b', re.I)
    pronouns = pronoun_count.findall(text)
    return len(pronouns)
```

In [52]:

```
avg_sent_length = []
percent_of_complex_words = []
fog_index = []
url_idss = []
no_of_complex_word=[]
word_count=[]
personal_pronouns_count = []
word_avg_length = []

import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize, sent_tokenize
path3=r'E:/Black_Coffer_Assignment/Web_Scrapped_Articles/'
for file in all_files1:
    url_idss.append(int(file.replace(".txt",'')))
    with open(os.path.join(path3, file),"rb") as q:
        text3 = q.read()
        word_tk=word_tokenize(str(text3))
        sent_tk=sent_tokenize(str(text3))
        avg_sent_length.append(len(word_tk)/len(sent_tk))
        percent_of_complex_words.append(count_complex_words(word_tk)/len(word_tk))
        fog_index.append(0.4*(len(word_tk)/len(sent_tk))+(count_complex_words(word_tk)/len(word_tk)))
        no_of_complex_word.append(count_complex_words(word_tk))
        word_count.append(len(word_tk))
    c = 0
    for word in word_tk:
        c += len(word)
    personal_pronouns_count.append(count_personal_pronouns(str(text3)))
    word_avg_length.append(round(c/len(word_tk)))
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\sudip\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

In [79]:

```
df2=pd.DataFrame({'URL_ID':url_idss,'AVG SENTENCE LENGTH': avg_sent_length,
                  'PERCENTAGE OF COMPLEX WORDS':percent_of_complex_words,
                 'FOG INDEX':fog_index, 'AVG NUMBER OF WORDS PER SENTENCE': avg_sent_length,
                 'COMPLEX WORD COUNT': no_of_complex_word, 'WORD COUNT': word_count})

df4=pd.DataFrame({'URL_ID':url_idss, 'PERSONAL PRONOUNS':personal_pronouns_count, 'AVG WORD LENGTH':word_avg_length})
```

In [91]:

```
output1=pd.merge(urls, df1, on = "URL_ID", how = "outer")
output2=pd.merge(output1, df2, on="URL_ID", how="outer")
output3=pd.merge(output2, df3, on="URL_ID", how="outer")
Final_Output=pd.merge(output3, df4, on="URL_ID", how="outer")
```

In [96]:

```
Final_Output
```

Out[96]:

	URL_ID	URL	POSITIVE SCORE	NEGATIVE SCORE	POLARITY SCORE	SUBJECTIVITY SCORE	AVG SENTENCE LENGTH	PERCENTAGE OF COMPLEX WORDS	FOG INDEX	S
0	37	https://insights.blackcoffer.com/ai-in-healthc...	66.0	34.0	0.320000	0.100100	26.626667	0.222834	10.873501	
1	38	https://insights.blackcoffer.com/what-if-the-c...	58.0	37.0	0.221053	0.163230	20.922078	0.150838	8.519669	
2	39	https://insights.blackcoffer.com/what-jobs-wil...	65.0	35.0	0.300000	0.117096	22.678571	0.208399	9.279828	
3	40	https://insights.blackcoffer.com/will-machine-...	66.0	28.0	0.404255	0.141353	19.763441	0.136017	8.041394	
4	41	https://insights.blackcoffer.com/will-ai-repla...	60.0	27.0	0.379310	0.104067	29.151515	0.175156	11.835762	
...	...	...	...	...	...	...	...	...	...	
109	146	https://insights.blackcoffer.com/blockchain-fo...	22.0	28.0	-0.120000	0.108225	20.387755	0.184184	8.339286	
110	147	https://insights.blackcoffer.com/the-future-of...	36.0	15.0	0.411765	0.063670	28.274194	0.159726	11.469404	
111	148	https://insights.blackcoffer.com/big-data-anal...	28.0	47.0	-0.253333	0.120773	20.580645	0.187304	8.419562	
112	149	https://insights.blackcoffer.com/business-anal...	35.0	7.0	0.666667	0.109375	31.120000	0.259640	12.707640	
113	150	https://insights.blackcoffer.com/challenges-an...	31.0	40.0	-0.126761	0.136015	17.439394	0.170287	7.146044	

114 rows × 15 columns

In [95]:

```
Final_Output.to_excel('E:\Black_Coffer_Assignment\Output Data Structure.xlsx', index=False)
```

In [ ]:

```

```