In [ ]:
```
##### import pandas as pd
import numpy as np
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import KFold
from sklearn.decomposition import TruncatedSVD
import pickle
import nltk
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.stem.snowball import SnowballStemmer
from nltk.corpus import stopwords
# nltk.download("stopwords")
# nltk.download("punkt")
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
import re
import string

from datetime import datetime
from sklearn.linear_model import LogisticRegression

from gensim.models import Word2Vec
```

In [40]:
```
# Convert the timestamps to day-month-year and real time
def timestamp_to_date(df):
    time_arr = []
    day_arr = []
    year_arr = []
    month_arr = []
    for val in df["Time"]:
        date_time = str(datetime.fromtimestamp(val))
        date = date_time.split(" ")[0]
        time = date_time.split(" ")[1]
        year = date.split("-")[0]
        month = date.split("-")[1]
        day = date.split("-")[2]
        time_arr.append(time)
        day_arr.append(day)
        month_arr.append(month)
        year_arr.append(year)
    df["Real_Time"] = np.array(time_arr)
    df["Year"] = np.array(year_arr).astype(int)
    df["Month"] = np.array(month_arr).astype(int)
    df["Day"] = np.array(day_arr).astype(int)

    return df
```

In [41]:
```
# Deal with text stemmer
# col: either "Text", or "Summary"
def text_stem(df, col):
    snowball_stemmer = SnowballStemmer(language='english')

    # Stemmed words
    stemmed_words = []
```

```
    for value in list(df[col]):
        tokenized_article = word_tokenize(value)

        stemmed_article = ''
        for j in range(len(tokenized_article)):
            word = snowball_stemmer.stem(tokenized_article[j])
            stemmed_article += " " + word

        stemmed_words.append(stemmed_article)

    df[f'{col}_Stemmed'] = np.array(stemmed_words)
    return df
```

In [42]:
```
X_t = pd.read_csv("./data/X_train.csv")
X_s = pd.read_csv("./data/X_test.csv")

X_train = X_t

X_train.shape
X_train.head()
```

Out[42]:

	Id	ProductId	UserId	HelpfulnessNumerator	HelpfulnessDenominator	T
0	195370	1890228583	A3VLX5Z090RQ0V	1	2	103083
1	1632470	B00BEIYSL4	AUDXDMFM49NGY	0	1	140503
2	9771	0767809335	A3LFIA97BUU5IE	3	36	98375
3	218855	6300215792	A1QZM75342ZQVQ	1	1	139484
4	936225	B000B5XOZW	ANM2SCEUL3WL1	1	1	116372

In [81]:
```
# Text Process Step
def process_sentence(df):
    alphanumeric = lambda x: re.sub(r"""\w*\d\w*""", ' ', x)
```

```
    punc_lower = lambda x: re.sub('[%s]' % re.escape(string.punctuation), ' ', x.lower

    # Reassign the process
    df["Text"] = df["Text"].fillna("").map(alphanumeric).map(punc_lower)
    df["Summary"] = df["Summary"].fillna("").map(alphanumeric).map(punc_lower)
    return df
```

In [44]:
```
X_train = process_sentence(X_train)
X_train = timestamp_to_date(X_train)
X_train.head()
```

Out[44]:

	Id	ProductId	UserId	HelpfulnessNumerator	HelpfulnessDenominator	T
0	195370	1890228583	A3VLX5Z090RQ0V	1	2	103083
1	1632470	B00BEIYSL4	AUDXDMFM49NGY	0	1	140503
2	9771	0767809335	A3LFIA97BUU5IE	3	36	98375
3	218855	6300215792	A1QZM75342ZQVQ	1	1	139484
4	936225	B000B5XOZW	ANM2SCEUL3WL1	1	1	116372

In [45]:
```
# Stemmed Text
X_train = text_stem(X_train, "Summary")
X_train = text_stem(X_train, "Text")

# X_train.shape
X_train.head()
```

Out[45]:

	Id	ProductId	UserId	HelpfulnessNumerator	HelpfulnessDenominator	T
0	195370	1890228583	A3VLX5Z090RQ0V	1	2	103083
1	1632470	B00BEIYSL4	AUDXDMFM49NGY	0	1	140503
2	9771	0767809335	A3LFIA97BUU5IE	3	36	98375
3	218855	6300215792	A1QZM75342ZQVQ	1	1	139484
4	936225	B000B5XOZW	ANM2SCEUL3WL1	1	1	116372

In [46]:
```
## Do the same for test_set
X_test = X_s
X_test = process_sentence(X_test)
X_test = timestamp_to_date(X_test)
# Stemmed Text
X_test = text_stem(X_test, "Summary")
X_test = text_stem(X_test, "Text")
# X_train.shape
X_test.head()
```

Out[46]:

	Id	ProductId	UserId	HelpfulnessNumerator	HelpfulnessDenominator	
0	786781	B0000VD02Y	A1UL8PS42M5DM8	1	7	10823
1	17153	0767823931	A2OP1HD9RGX5OW	3	6	10553
2	1557328	B008JFUNTG	AY113687D8YK1	1	8	13773
3	1242666	B001UWOLQG	A2MVTAEGBP08RB	0	1	13747
4	1359242	B003QS0E54	ALGAE0IGE4DBP	99	103	12766