In [1]:
```python
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import sqlite3
import csv
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import re
import os
from sqlalchemy import create_engine # database connection
import datetime as dt
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import SGDClassifier
from sklearn import metrics
from sklearn.metrics import f1_score,precision_score,recall_score
from sklearn import svm
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from datetime import datetime

from tqdm import tqdm
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.metrics import pairwise_distances
```

In [2]:
```python
data_main_clean_v4=pd.read_pickle('data_main_clean_v4.pickle')
```

In [3]:
```python
data_main_clean_v4
```

Out[3]:

	index	Title	Tokens	Cleaned_Title	Title_Length
0	0	implementing boundary value analysis of softwa...	[c++, testing]	implementing boundary value analysis software ...	74
1	2	java.lang.noclassdeffounderror: javax/servlet/...	[java, jsp]	java lang noclassdeffounderror javax servlet j...	76
2	3	java.sql.sqlexception: [microsoft][odbc driver ...	[java, sql]	java sql sqlexception microsoft odbc driver ma...	79
3	4	better way to update feed on fb with php sdk	[php]	better way update feed fb php sdk	44
4	6	"sql injection" issue preventing correct form ...	[php, sql]	sql injection issue preventing correct form su...	62
...	...	...	...	...	...
1396265	3999980	scrolling issue of uitableview and web view	[uitableview]	scrolling issue uitableview web view	43
1396266	3999981	scrolling issue while adding heavy file in uiw...	[iphone, file]	scrolling issue adding heavy file uiwebview ip...	62
1396267	3999986	scrolling list view color change in android	[android, list]	scrolling list view color change android	43
1396268	3999988	scrolling listview causes buttons to be invisible	[listview]	scrolling listview causes buttons invisible	49
1396269	3999996	scrolling on touch devices for phonegap/cordov...	[phonegap]	scrolling touch devices phonegap cordova projects	56

1396270 rows × 5 columns

BOW

BOW Default Parameters

In [4]:
```
vectorizer_bow = CountVectorizer()
text_bow = vectorizer_bow.fit_transform(data_main_clean_v4['Cleaned_Tit
le'].values)
text_bow.shape
```

Out[4]:
```
(1396270, 118052)
```

In [289]:
```
def Recomend(string):

    #Preprocessing the input string in real time
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~')) #This is used since we want the whole sen
tenence in a single list

    #Tokenizing the input in n-dim array
    process=vectorizer_bow.transform(a)
    query=process.toarray()
    #Finding distances of the entered point from all the points
    distance  = pairwise_distances(text_bow, query.reshape(1,-1),metric
='cosine')
    indices = np.argsort(distance.flatten())[0:10] #Returning top 10 le
ast distance indices
    pdists  = np.sort(distance.flatten())[0:10] #Returning top 10 least
 distances
    print('RECOMENDED SIMILAR QUESTIONS')

    #Prinitng the points which have the lowest distance
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(distance[i])),
4))
        print('\n')
```

In [290]:

```
import time
start_time = time.time()
Recomend('implementing boundary value analysis software testing c++ pro
gram')
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.4331
```

```
3 th question " equivalence class testing vs boundary value testing "
3 th question distance is  0.496


4 th question " using log analysis tools software testing "
4 th question distance is  0.5371


5 th question " boundary value analysis string values date "
5 th question distance is  0.5371


6 th question " types software testing "
6 th question distance is  0.5636


7 th question " implementing shell c program "
7 th question distance is  0.5636


8 th question " p-value 0 testing distribution "
8 th question distance is  0.5636


9 th question " software testing tool requirements testing "
9 th question distance is  0.5714


10 th question " software testing domains testing skills "
10 th question distance is  0.5714


TIME TAKEN TO FETCH RESULTS
0.6512563228607178 seconds
```

BOW N-Gram with max_features of 20000

In [155]:
```
vectorizer_bow_v2 = CountVectorizer(max_features=20000,ngram_range=(1,3
))
text_bow_v2 = vectorizer_bow_v2.fit_transform(data_main_clean_v4['Clean
ed_Title'].values)
```

In [156]:
```
def Recomend(string):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~'))
    process=vectorizer_bow_v2.transform(a)
    query=process.toarray()
    distance  = pairwise_distances(text_bow_v2, query.reshape(1,-1),met
ric='cosine')
    indices = np.argsort(distance.flatten())[0:10]
    pdists  = np.sort(distance.flatten())[0:10]
    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
```

```
        print(g ,'th question distance is ',round((float(distance[i])),
4))
        print('\n')
```

In [157]:
```
Recomend('implementing boundary value analysis software testing c++ pro
gram')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.3453


3 th question " serendipity booksellers software program c++ "
3 th question distance is  0.4655


4 th question " equivalence class testing vs boundary value testing "
4 th question distance is  0.496


5 th question " using log analysis tools software testing "
5 th question distance is  0.5371


6 th question " testing circuit implementing kruskalls algorithm "
6 th question distance is  0.5636


7 th question " p-value 0 testing distribution "
7 th question distance is  0.5636
```

```
8 th question " meaning incident software testing "
8 th question distance is  0.5636


9 th question " source statistic effectiveness software testing "
9 th question distance is  0.5636


10 th question " implementing shell c program "
10 th question distance is  0.5636
```

BOW with N_Gram and token features

Trying to do somthing like weighted average by giving more weights to tokens.

In [299]:
```
vectorizer_bow_v3 = CountVectorizer(max_features=20000,ngram_range=(1,3
))
text_bow_v3 = vectorizer_bow_v3.fit_transform(data_main_clean_v4['Clean
ed_Title'].values)
```

In [217]:
```
#Converting lists in space seperated strings so that it can be used for
 vectorizing
a1=[]
for i in range(0,len(data_main_clean_v3)):
    a1.append(' '.join(data_main_clean_v3['Tokens'][i]))
```

In [220]:
```
data_main_clean_v3['Token_Space']=a1
data_main_clean_v4=data_main_clean_v3
data_main_clean_v4.to_pickle('data_main_clean_v4.pickle')
```

In [236]:
```
data_main_clean_v4.head()
```

Out[236]:

	index	Title	Tokens	Cleaned_Title	Title_Length	Token_Space
0	0	implementing boundary value analysis of softwa...	[c++, testing]	implementing boundary value analysis software ...	74	c++ testing
1	2	java.lang.noclassdeffounderror: javax/servlet/...	[java, jsp]	java lang noclassdeffounderror javax servlet j...	76	java jsp
2	3	java.sql.sqlexception: [microsoft][odbc driver ...	[java, sql]	java sql sqlexception microsoft odbc driver ma...	79	java sql
3	4	better way to update feed on fb with php sdk	[php]	better way update feed fb php sdk	44	php
4	6	"sql injection" issue preventing correct form ...	[php, sql]	sql injection issue preventing correct form su...	62	php sql

Vectorizing tokens

In [239]:
```
token_vec = CountVectorizer(tokenizer = lambda x: x.split())
token  = token_vec.fit_transform(data_main_clean_v4['Token_Space'].valu
es)
```

In [300]:
```
def Recomend_1(string,token_weight,text_weight):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~'))
    process=vectorizer_bow_v3.transform(a)
    query=process.toarray()
    distance  = pairwise_distances(text_bow_v3, query.reshape(1,-1),met
ric='cosine')
    print(sent_1)

    # Finding that if there are any tokens in the user input in real ti
me
    bb=[]
    qq=sent_1.split()
    for i in qq:
        for j in token_vec.get_feature_names():
            if (i==j):
                bb.append(i)
    bb=' '.join(bb)
    bb=list(bb.split('~'))

    #Transforming tokens in real time
    tokens_transform=token_vec.transform(bb)
    tok=tokens_transform.toarray()
    tok_dist=pairwise_distances(token, tok.reshape(1,-1),metric='cosin
e')


    # Taking weighted measure of text and tokens
    final=(token_weight*tok_dist+text_weight*distance)/float(text_weigh
t+token_weight)

    # Returning with lowest distance
    indices = np.argsort(final.flatten())[0:10]
    pdists  = np.sort(final.flatten())[0:10]
    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(final[i])),4))
        print('\n')
```

In [312]:

```
import time
start_time = time.time()
Recomend_1('implementing boundary value analysis software testing c++ p
rogram',10,30)
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


implementing boundary value analysis software testing c++ program
RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.3322


3 th question " serendipity booksellers software program c++ "
3 th question distance is  0.4223


4 th question " equivalence class testing vs boundary value testing "
4 th question distance is  0.4639
```

```
5 th question " using log analysis tools software testing "
5 th question distance is  0.476


6 th question " scoring analysis subjective testing skills assessment "
6 th question distance is  0.4959


7 th question " justify software testing management "
7 th question distance is  0.4959


8 th question " meaning incident software testing "
8 th question distance is  0.4959


9 th question " types software testing "
9 th question distance is  0.4959


10 th question " cpu health testing software "
10 th question distance is  0.4959


TIME TAKEN TO FETCH RESULTS
0.6153523921966553 seconds
```

Bow Default features + Tokens

In [291]:

```
vectorizer_bow_v4 = CountVectorizer()
text_bow_v4 = vectorizer_bow_v4.fit_transform(data_main_clean_v4['Clean
ed_Title'].values)
```

In [394]:

```
def Recomend_1(string,token_weight,text_weight):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~'))
    process=vectorizer_bow_v4.transform(a)
    query=process.toarray()
    distance  = pairwise_distances(text_bow_v4, query.reshape(1,-1),met
ric='cosine')
    print(sent_1)
    print(type(sent_1))


    bb=[]
    qq=sent_1.split()
    for i in qq:
        for j in token_vec.get_feature_names():
            if (i==j):
                bb.append(i)
    bb=' '.join(bb)
    bb=list(bb.split('~'))
    tokens_transform=token_vec.transform(bb)
    tok=tokens_transform.toarray()
    tok_dist=pairwise_distances(token, tok.reshape(1,-1),metric='cosin
e')
```

```
    final=(token_weight*tok_dist+text_weight*distance)/float(text_weigh
t+token_weight)


    indices = np.argsort(final.flatten())[0:10]
    pdists  = np.sort(final.flatten())[0:10]
    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(final[i])),4))
        print('\n')
```

In [395]:

```
import time
start_time = time.time()
Recomend_1('implementing boundary value analysis software testing c++ p
rogram',10,40)
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


implementing boundary value analysis software testing c++ program
<class 'str'>
RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.405
```

```
3 th question " equivalence class testing vs boundary value testing "
3 th question distance is  0.4703


4 th question " using log analysis tools software testing "
4 th question distance is  0.4883


5 th question " types software testing "
5 th question distance is  0.5094


6 th question " p-value 0 testing distribution "
6 th question distance is  0.5094


7 th question " static analysis dynamic analysis testing "
7 th question distance is  0.5157


8 th question " choose software development software testing "
8 th question distance is  0.5157


9 th question " software testing domains testing skills "
9 th question distance is  0.5157


10 th question " software testing tool requirements testing "
10 th question distance is  0.5157


TIME TAKEN TO FETCH RESULTS
0.5532786846160889 seconds
```

Summary from BOW

1. All the models are working fair enough and are able to return results withing 1 second
2. Model after giving more token weight is successfull in bringing results with same token
3. Out of 4 models, the third model with Weighted Token and N_Gram is working good

Limitations:

1. Not considering semantic meaning of words

TFIDF

TDIDF with default parameters

In [411]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer_tfidf = TfidfVectorizer()
text_tfidf = vectorizer_tfidf.fit_transform(data_main_clean_v4['Cleaned_Title'].values)
text_tfidf.shape
```

Out[411]:
```
(1396270, 118052)
```

In [412]:
```
def Recomend(string):

    #Preprocessing the input string in real time
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~')) #This is used since we want the whole sen
tenence in a single list

    #Tokenizing the input in n-dim array
    process=vectorizer_tfidf.transform(a)
    query=process.toarray()

    #Finding distances of the entered point from all the points
    distance  = pairwise_distances(text_tfidf, query.reshape(1,-1),metr
ic='cosine')
    indices = np.argsort(distance.flatten())[0:10] #Returning top 10 le
ast distance indices
    pdists  = np.sort(distance.flatten())[0:10] #Returning top 10 least
 distances
    print('RECOMENDED SIMILAR QUESTIONS')

    #Prinitng the points which have the lowest distance
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(distance[i])),
4))
        print('\n')
```

In [414]:
```
import time
```

```python
start_time = time.time()
Recomend('implementing boundary value analysis software testing c++ pro
gram')
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis string values date "
2 th question distance is  0.392


3 th question " boundary value analysis c++ cppunit "
3 th question distance is  0.4601


4 th question " using log analysis tools software testing "
4 th question distance is  0.4855


5 th question " equivalence class testing vs boundary value testing "
5 th question distance is  0.5045


6 th question " static analysis dynamic analysis testing "
6 th question distance is  0.5305


7 th question " implementing boundary-fill algorithm opengl "
7 th question distance is  0.5438
```

```
8 th question " implementing java analysis algorithms "

8 th question distance is  0.5753


9 th question " types software testing "
9 th question distance is  0.5936


10 th question " implementing shell c program "
10 th question distance is  0.5975


TIME TAKEN TO FETCH RESULTS
0.40691137313842773 seconds
```

TFIDF with N_Gram

In [415]:

```
vectorizer_tfidf_v2 = TfidfVectorizer(max_features=20000,ngram_range=(1
,3))
text_tfidf_v2 = vectorizer_tfidf_v2.fit_transform(data_main_clean_v4['C
leaned_Title'].values)
```

In [416]:

```
def Recomend(string):

    #Preprocessing the input string in real time
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~')) #This is used since we want the whole sen
tenence in a single list

    #Tokenizing the input in n-dim array
    process=vectorizer_tfidf_v2.transform(a)
    query=process.toarray()

    #Finding distances of the entered point from all the points
    distance  = pairwise_distances(text_tfidf_v2, query.reshape(1,-1),m
etric='cosine')
    indices = np.argsort(distance.flatten())[0:10] #Returning top 10 le
ast distance indices
    pdists  = np.sort(distance.flatten())[0:10] #Returning top 10 least
 distances
    print('RECOMENDED SIMILAR QUESTIONS')

    #Prinitng the points which have the lowest distance
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(distance[i])),
4))
        print('\n')
```

In [417]:
```
import time
start_time = time.time()
Recomend('implementing boundary value analysis software testing c++ pro
gram')
```

```python
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.2815


3 th question " boundary value analysis string values date "
3 th question distance is  0.4752


4 th question " using log analysis tools software testing "
4 th question distance is  0.4855


5 th question " equivalence class testing vs boundary value testing "
5 th question distance is  0.5045


6 th question " serendipity booksellers software program c++ "
6 th question distance is  0.5113


7 th question " static analysis dynamic analysis testing "
7 th question distance is  0.5305


8 th question " implementing boundary-fill algorithm opengl "
8 th question distance is  0.5438
```

```
9 th question " find boundary mathcal c 1 manifold "

9 th question distance is  0.5459


10 th question " um modeling analysis class boundary vs control class "
10 th question distance is  0.5678


TIME TAKEN TO FETCH RESULTS
0.41884326934814453 seconds
```

TFIDF with ngram and weights to token features

In [425]:
```
vectorizer_tfidf_v3 = TfidfVectorizer(max_features=20000,ngram_range=(1
,3))
text_tfidf_v3 = vectorizer_tfidf_v3.fit_transform(data_main_clean_v4['C
leaned_Title'].values)
```

In [419]:
```
def Recomend_1(string,token_weight,text_weight):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~'))
    process=vectorizer_tfidf_v3.transform(a)
    query=process.toarray()
    distance  = pairwise_distances(text_tfidf_v3, query.reshape(1,-1),m
etric='cosine')
    print(sent_1)

    # Finding that if there are any tokens in the user input in real ti
me
    bb=[]
    qq=sent_1.split()
    for i in qq:
        for j in token_vec.get_feature_names():
            if (i==j):
                bb.append(i)
    bb=' '.join(bb)
    bb=list(bb.split('~'))

    #Transforming tokens in real time
    tokens_transform=token_vec.transform(bb)
    tok=tokens_transform.toarray()
    tok_dist=pairwise_distances(token, tok.reshape(1,-1),metric='cosin
e')


    # Taking weighted measure of text and tokens
    final=(token_weight*tok_dist+text_weight*distance)/float(text_weigh
t+token_weight)

    # Returning with lowest distance
    indices = np.argsort(final.flatten())[0:10]
    pdists  = np.sort(final.flatten())[0:10]
    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(final[i])),4))
        print('\n')
```

In [424]:

```
import time
start_time = time.time()
Recomend_1('implementing boundary value analysis software testing c++ p
rogram',10,40)
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


implementing boundary value analysis software testing c++ program
RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.2838


3 th question " using log analysis tools software testing "
3 th question distance is  0.447


4 th question " serendipity booksellers software program c++ "
4 th question distance is  0.4676


5 th question " equivalence class testing vs boundary value testing "
5 th question distance is  0.4771


6 th question " static analysis dynamic analysis testing "
```

```
6 th question distance is  0.483


7 th question " testing multiple regexps time use syntactic analysis "
7 th question distance is  0.5202


8 th question " source statistic effectiveness software testing "
8 th question distance is  0.5228


9 th question " types software testing "
9 th question distance is  0.5335


10 th question " justify software testing management "
10 th question distance is  0.5397


TIME TAKEN TO FETCH RESULTS
0.5236008167266846 seconds
```

TFIDF Default parameters and token features

In [427]:
```
vectorizer_tfidf_v4 = TfidfVectorizer()
text_tfidf_v4 = vectorizer_tfidf_v4.fit_transform(data_main_clean_v4['C
leaned_Title'].values)
```

In [435]:
```
def Recomend_1(string,token_weight,text_weight):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')
    a=list(sent_1.split('~'))
    process=vectorizer_tfidf.transform(a)
    query=process.toarray()
    distance  = pairwise_distances(text_tfidf, query.reshape(1,-1),metr
ic='cosine')
    print(sent_1)

    # Finding that if there are any tokens in the user input in real ti
me
    bb=[]
    qq=sent_1.split()
    for i in qq:
        for j in token_vec.get_feature_names():
            if (i==j):
                bb.append(i)
    bb=' '.join(bb)
    bb=list(bb.split('~'))

    #Transforming tokens in real time
    tokens_transform=token_vec.transform(bb)
    tok=tokens_transform.toarray()
    tok_dist=pairwise_distances(token, tok.reshape(1,-1),metric='cosin
e')


    # Taking weighted measure of text and tokens
    final=(token_weight*tok_dist+text_weight*distance)/float(text_weigh
t+token_weight)

    # Returning with lowest distance
    indices = np.argsort(final.flatten())[0:10]
    pdists  = np.sort(final.flatten())[0:10]
    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(final[i])),4))
        print('\n')
```

In [436]:
```
import time
start_time = time.time()
Recomend_1('implementing boundary value analysis software testing c++ p
rogram',10,40)
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


implementing boundary value analysis software testing c++ program
RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " boundary value analysis c++ cppunit "
2 th question distance is  0.4266


3 th question " using log analysis tools software testing "
3 th question distance is  0.447
```

```
4 th question " equivalence class testing vs boundary value testing "
4 th question distance is  0.4771


5 th question " static analysis dynamic analysis testing "
5 th question distance is  0.483


6 th question " boundary value analysis string values date "
6 th question distance is  0.5136


7 th question " types software testing "
7 th question distance is  0.5335


8 th question " choose software development software testing "
8 th question distance is  0.5412


9 th question " software testing tools testing web application "
9 th question distance is  0.5488


10 th question " unit testing tools generating boundary conditions "
10 th question distance is  0.5553


TIME TAKEN TO FETCH RESULTS
0.5006287097930908 seconds
```

Observations:

1. Getting good performance and similar queries on N-Grams (BI-Grams)
2. Using token features to get resuts of similar tokens

TFIDF W2V

In [314]:
```
import pickle
with open('glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words = set(model.keys())
```

In [315]:
```
avg_w2v_vectors_train = []; # the avg-w2v for each sentence/review is s
tored in this list
for sentence in tqdm(data_main_clean_v4['Cleaned_Title'].values): # for
 each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/re
view
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_vectors_train.append(vector)
```

```
100%|█████████████████████████████████████████████████████████████████
███| 1396270/1396270 [00:23<00:00, 58345.57it/s]
```

In [321]:
```
avg_w2v_vectors_train_1=np.array(avg_w2v_vectors_train)
```

In [322]:
```
avg_w2v_vectors_train_1.shape
```

Out[322]:
```
(1396270, 300)
```

In [382]:
```
def Recomend(string):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')


    a=sent_1

    vector_1 = np.zeros(300)
    cnt_words_1 =0;
    for word in a.split(): # for each word in a review/sentence
        if word in glove_words:
            vector_1 += model[word]
            cnt_words_1 += 1
    if cnt_words_1 != 0:
        vector_1 /= cnt_words_1


    query=vector_1
    distance  = pairwise_distances(avg_w2v_vectors_train_1[0:1000000],
query.reshape(1,-1),metric='cosine')
    indices = np.argsort(distance.flatten())[0:10]
    pdists  = np.sort(distance.flatten())[0:10]

    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(distance[i])),
4))
        print('\n')
```

In [384]:
```
import time
start_time = time.time()
Recomend('implementing boundary value analysis software testing c++ pro
gram')
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0


2 th question " justify software testing management "
2 th question distance is  0.0968


3 th question " compatibility test testing method use building software
"
3 th question distance is  0.1001


4 th question " rest client software development testing "
4 th question distance is  0.1028


5 th question " automated testing explaining business value "
5 th question distance is  0.1031
```

```
6 th question " best unit testing framework testing wp7 application "
6 th question distance is  0.1067


7 th question " categorise various software testing methods "
7 th question distance is  0.1082


8 th question " need idea source code testing evaluation tool "
8 th question distance is  0.11


9 th question " best practice data validation enterprise application "
9 th question distance is  0.1107


10 th question " extending homework testing platform include code analy
sis c c++ "
10 th question distance is  0.1109


TIME TAKEN TO FETCH RESULTS
4.295557975769043 seconds
```

TFIDF + TOKEN Weighted

In [401]:
```
def Recomend_1(string,token_weight,text_weight):
    stopwords_1 = stopwords.words("english")
    a=string
    sent_1=a.lower().strip()
    sent_1 = re.sub(r"won\'t", "will not", sent_1)
    sent_1 = re.sub(r"can\'t", "can not", sent_1)
    sent_1 = re.sub(r"n\'t", " not", sent_1)
    sent_1 = re.sub(r"\'re", " are", sent_1)
    sent_1 = re.sub(r"\'s", " is", sent_1)
    sent_1 = re.sub(r"\'d", " would", sent_1)
    sent_1 = re.sub(r"\'ll", " will", sent_1)
    sent_1 = re.sub(r"\'t", " not", sent_1)
    sent_1 = re.sub(r"\'ve", " have", sent_1)
    sent_1 = re.sub(r"\'m", " am", sent_1)
    sent_1 = re.sub('[^A-Za-z0-9-+]+', ' ', sent_1)
    sent_1 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_1=sent_1.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(sent_1)
    print('\n')



    a=sent_1

    vector_1 = np.zeros(300)
    cnt_words_1 =0;
    for word in a.split(): # for each word in a review/sentence
        if word in glove_words:
            vector_1 += model[word]
            cnt_words_1 += 1
    if cnt_words_1 != 0:
        vector_1 /= cnt_words_1



    query=vector_1
    distance  = pairwise_distances(avg_w2v_vectors_train_1[0:1000000],
query.reshape(1,-1),metric='cosine')



    bb=[]
    qq=sent_1.split()
    for i in qq:
        for j in token_vec.get_feature_names():
            if (i==j):
                bb.append(i)
    bb=' '.join(bb)
    bb=list(bb.split('~'))
    tokens_transform=token_vec.transform(bb)
    tok=tokens_transform.toarray()
    tok_dist=pairwise_distances(token[0:1000000], tok.reshape(1,-1),met
ric='cosine')



    final=(token_weight*tok_dist+text_weight*distance)/float(text_weigh
t+token_weight)



    indices = np.argsort(distance.flatten())[0:10]
    pdists  = np.sort(distance.flatten())[0:10]

    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v4['Cleaned_Title'][
i],'"')
        print(g ,'th question distance is ',round((float(final[i])),4))
        print('\n')
```

In [405]:
```
import time
start_time = time.time()
Recomend_1('implementing boundary value analysis software testing c++ p
rogram',50,10)
print('TIME TAKEN TO FETCH RESULTS')
print(time.time()-start_time,'seconds')
```

```
QUERY ENTERED BY THE USER
implementing boundary value analysis software testing c++ program


RECOMENDED SIMILAR QUESTIONS
1 th question " implementing boundary value analysis software testing c
++ program "
1 th question distance is  0.0
```

```
2 th question " justify software testing management "
2 th question distance is  0.2602


3 th question " compatibility test testing method use building software
"
3 th question distance is  0.2608


4 th question " rest client software development testing "
4 th question distance is  0.4338


5 th question " automated testing explaining business value "
5 th question distance is  0.2613


6 th question " best unit testing framework testing wp7 application "
6 th question distance is  0.2619


7 th question " categorise various software testing methods "
7 th question distance is  0.2621


8 th question " need idea source code testing evaluation tool "
8 th question distance is  0.2624


9 th question " best practice data validation enterprise application "
9 th question distance is  0.8518


10 th question " extending homework testing platform include code analy
sis c c++ "
10 th question distance is  0.2626
```

```
TIME TAKEN TO FETCH RESULTS
6.461676836013794 seconds
```

Observation:

1. Time taken to compute distance is bit higher in Word To Vec models.
2. Considering the sematic meaning of words like justifying and implement

In [ ]:

```
```

In [ ]:

```
```