In [8]:
```
#Using Universal Sentence Encoder with cosine distance
def Recomend(string):

    #Importing Required Modules
    import numpy as np
    from nltk.corpus import stopwords
    import warnings
    warnings.filterwarnings("ignore")
    import pandas as pd
    import numpy as np
    import re
    from nltk.corpus import stopwords
    from sklearn.multiclass import OneVsRestClassifier
    from sklearn.linear_model import SGDClassifier
    from datetime import datetime
    from sklearn.metrics.pairwise import cosine_similarity
    from sklearn.metrics import pairwise_distances
    from numpy import load

    #EDA
    data_main_clean_v5=pd.read_pickle('data_main_clean_v5.pickle') #Loa
ding the cleaned and preprocessed data


    #Preprocessing text and removing stopwords
    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_2 = ' '.join(e for e in sent_1.split() if e not in stopwords_1
)
    sent_2=sent_2.lower().strip()
    print('QUERY ENTERED BY THE USER')
    print(a)
    print('\n')

    #TAKES A LONG TIME DONT RUN. This part of code return 512 dimesiona
l embeddings for all 1M data points

    #import tensorflow_hub as hub
    #module_url = "https://tfhub.dev/google/universal-sentence-encoder-
large/3"
    #embed = hub.KerasLayer(module_url)
    ###Find embeddings using API
    #import numpy as np
    #import tensorflow as tf
    #np_list = np.asarray(data_main_clean_v5['Cleaned_Title'])
    #tensor_list = tf.convert_to_tensor(np_list)
    #a=list(tensor_list)
    #embedding = embed(tensor_list)
    #z=np.array(embedding)
    ###Save embeddings to box
    #from numpy import savez_compressed
    #savez_compressed('USE_EMBEDDINGS', z)

    #I have already saved the output of above code and have imported th
e same below as 'USE_EMBEDDINGS.npz'

    dict_data = load('USE_EMBEDDINGS.npz') #Importing 512 dimesional em
beddings for all data-points
    a1 = dict_data['arr_0'] #a1 contains the embedding vectors

    #Using Google API to convert user entered query to 512 dimensional
 vector in real time
    #Universal Sentence Encoder
```

```
    module_url = "https://tfhub.dev/google/universal-sentence-encoder-large/3" #URL
    embed = hub.KerasLayer(module_url) #Embedding with the above URL 512 dimensional
    a2=[sent_1] #Converting query to a list
    np_list = np.asarray(a2) #Converting list to numpy array
    tensor_list = tf.convert_to_tensor(np_list) #Converting numpy list to tensor as embedding requires tensor list
    embedding = embed(tensor_list) #Embedding the query to return 512 dimensional vector
    query=np.array(embedding) #COnverting the tensor to a numpy array so that we can perform numpy operations on it


    distance  = pairwise_distances(a1, query.reshape(1,-1),metric='cosine') #Finding cosine distance for entered query and all points
    indices = np.argsort(distance.flatten())[0:10] #Returning indices of top 10 lowest distance
    pdists  = np.sort(distance.flatten())[0:10] #Returning distances of top 10 nearest points

    print('RECOMENDED SIMILAR QUESTIONS')
    g=0
    # Code to print the top 10 closest points
    for i in indices:
        g=g+1
        print(g ,'th question','"',data_main_clean_v5['Cleaned_Title'][i],'"')
        print(g ,'th question distance is ',round((float(distance[i])),4))
        print('\n')
```

In [27]:
```
Recomend('how to create a linked list in python')
```

```
QUERY ENTERED BY THE USER
how to create a linked list in python


RECOMENDED SIMILAR QUESTIONS
```

```
1 th question " creating a python list from a list of tuples "
1 th question distance is  0.0485


2 th question " create a list of tuples from list in python "
2 th question distance is  0.0499


3 th question " python create list from specific indexes in a list of l
ists "
3 th question distance is  0.0501


4 th question " how do i extend a list within a list in python? "
4 th question distance is  0.0508


5 th question " make python sublists from a list using a seperator "
5 th question distance is  0.0533


6 th question " creating a list of methods to be executed in python "
6 th question distance is  0.0539


7 th question " python: how to create a new list based on existing list
without certain objects "
7 th question distance is  0.054


8 th question " how to convert python multiple list to list "
8 th question distance is  0.0545


9 th question " create multidimensional list in python through two othe
r lists "
9 th question distance is  0.0548
```

```
10 th question " python: make new tuple by attaching info from existing
list "
10 th question distance is  0.0564


Time required to fetch results 3.12865532875061 seconds
```