In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
# Code to read csv file into Colaboratory:
!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
# Authenticate and create the PyDrive client.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
```

In [3]:
```
import pandas as pd
downloaded = drive.CreateFile({'id':'1JnT4GeVfQ6IdVNMiQY265vc9RO42K6NN'})
downloaded.GetContentFile('Data.csv')
downloaded = drive.CreateFile({'id': '1pIf4tm-zbck_kLYkroa_a6vftB5KxTxS'})
downloaded.GetContentFile('glove_vectors')

downloaded = drive.CreateFile({'id': '1rEXLUijrOxF3eKJXJSv2jx2SgjOR0mLQ'})
downloaded.GetContentFile('case_study_1_M3.hdf5')
```

In [4]:
```
downloaded = drive.CreateFile({'id': '1EIoVy-bHEaaZ9rTlXBAm59rITbGzQBII'})
downloaded.GetContentFile('tokenizer.pkl')

downloaded = drive.CreateFile({'id': '1gfElSWL6ZGtIty_3LoT_agvqwZXWiOmF'})
downloaded.GetContentFile('vectorizer.pkl')
```

In [5]:
```
from numpy import array
from numpy import asarray
from numpy import zeros
# import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Embedding
from tensorflow.keras.layers import Input, Embedding, LSTM, Dense, concatenate, D
from tensorflow.keras.models import Model
from sklearn.metrics import f1_score
```

In [ ]:
```
Data = pd.read_csv("Data.csv")
```

In [ ]:
```
Data = Data.fillna("")
```

In [ ]:
```
Data.head(2)
```

Out[32]:

	questions_id	questions_title	questions_body	
0	0003e7bf48f24b5c985f8fce96e611f3	double major tech academy high school worth	junior h right thinking double major tech academy guaranteed internship thinking worth two tech classes back back whole senior year take college classes nearest community college know fact not soo much time give something not gonna help worth information technology majoring technology internship time high school	
1	0006609dd4da40dcaa5a83e0499aba14	declare minor undergrad want lawyer	currently undergrad want go law school lawyer thinking minoring psychology would helpful long run psychology law	business,career,careerpath,coll

In [ ]:
```
Data['Complete'] = Data['questions_title'].map(str) + " " + Data['questions_body'
Data.drop(['questions_title','questions_body','questions_id'], axis=1, inplace=Tr
```

In [ ]:
```
Data.head(2)
```

Out[9]:

	tags_tag_name	professionals_industry	Complete
0	cybersecurity,disciplined,none,tech,womentech	career_counseling computer_software	double major tech academy high school worth ju...
1	business,career,careerpath,college,collegeadmi...	insurance_law professional_training	declare minor undergrad want lawyer currently ...

In [ ]:
```
import regex as re
Data['tags_tag_name'] = Data['tags_tag_name'].str.replace(",", " ")
```

In [ ]:
```
Data.head(2)
```

Out[11]:

	tags_tag_name	professionals_industry	Complete
0	cybersecurity disciplined none tech womentech	career_counseling computer_software	double major tech academy high school worth ju...
1	business career careerpath college collegeadmi...	insurance_law professional_training	declare minor undergrad want lawyer currently ...

In [ ]:
```
X = Data.drop(['professionals_industry'], axis=1)
y = Data["professionals_industry"]
```

In [ ]:
```
y
```

Out[13]:
```
0                      career_counseling computer_software
1                      insurance_law professional_training
2                                                 research
3                    professional_trainer writing_editing
4                           financial_services_education
                              ...
23926
23927                                             zoo_keeping
23928    human_resources information_technology_service...
23929                                  education_management
23930            accounting_education education_management
Name: professionals_industry, Length: 23931, dtype: object
```

In [ ]:
```

```

In [ ]:
```
X.head(5)
```

Out[14]:

	tags_tag_name	Complete
0	cybersecurity disciplined none tech womentech	double major tech academy high school worth ju...
1	business career careerpath college collegeadmi...	declare minor undergrad want lawyer currently ...
2	brazil children coaching distanceeducation ngo...	get job prefered field degree next biology marine
3	editing instructionaldesign professionaltraini...	demand gym teachers diminishing hi currently d...
4	education educator financialservices generalin...	aspiring mathematician stay motivated research...

In [ ]:
```
print(len(pd.unique(Data['professionals_industry'])))
```

```
8016
```

In [ ]:
```
print(pd.unique(Data['professionals_industry']))
```

```
['career_counseling computer_software'
 ' insurance_law professional_training' 'research' ...
 'accounting primary_secondary_education telecommunications'
 'insurance materials_science_industrial_engineering pharmaceuticals'
 'human_resources information_technology_services_franchise_owner_commercial_cl
eaning_business_recruiter_career_counselor']
```

In [ ]:
```
Class_labels = pd.unique(Data['professionals_industry'])
Class_labels
```

Out[17]:
```
array(['career_counseling computer_software',
       ' insurance_law professional_training', 'research', ...,
       'accounting primary_secondary_education telecommunications',
       'insurance materials_science_industrial_engineering pharmaceuticals',
       'human_resources information_technology_services_franchise_owner_commerc
ial_cleaning_business_recruiter_career_counselor'],
      dtype=object)
```

In [ ]:
```
print(len(pd.unique(X.Complete)))
```

```
23877
```

In [ ]:
```
X.shape
```

Out[19]:
```
(23931, 2)
```

In [ ]:
```

```

In [ ]:
```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
y = vectorizer.fit_transform(y)
```

In [ ]:
```
feature_names = vectorizer.get_feature_names()
```

In [ ]:
```
print(len(feature_names))
```

```
1065
```

In [ ]:
```
print(feature_names)
```

```
['academia', 'academic_advising', 'academic_advisor', 'academic_websites', 'a
cademics', 'accounting', 'accounting_auditing', 'accounting_auditing_learning
_development', 'accounting_broker_dealers_hedge_funds_private_equity', 'accou
nting_commerce_financ_e', 'accounting_education', 'accounting_finance', 'acco
unting_information_technology', 'accounting_tax_operations_technology', 'acti
ng_motion_pictures_film', 'actor', 'administration', 'administrative', 'admin
istrative_assistance', 'administrative_teaching', 'advancement_engagement',
'advertising', 'advertising_analytics', 'advertising_photography', 'advertisi
ng_technology', 'advisory', 'aerospace', 'affordable_housing', 'agency_counse
ling_matriculating_cgs_lmhc_broadcast_media', 'agriculture', 'airbnb', 'aircr
aft_maintenance', 'airline_aviation', 'airlines_aviation', 'airlines_aviation
_hospitality', 'airlines_aviation_sales_artist_art_museum_chicago_parent_visu
al_designer', 'alternative_medicine', 'analytical_instruments_medical_device
s', 'analytics', 'analytics_technology', 'animal_behavior_welfare', 'animal
s', 'animation', 'animation_cartooning', 'animation_vfx_video_edition', 'appa
rel_fashion', 'apparel_fashion_fitness', 'archaeology', 'architecture_civil_e
ngineering', 'architecture_interior_architecture_interior_design', 'architect
ure_interior_design', 'architecture_planning', 'arco_gas_station', 'army', 'a
rt', 'art_blogging_caregiving', 'art_education', 'art_entertainment_product_d
```

In [ ]:
```
fg = pd.DataFrame(feature_names)
```

In [ ]:
```
fg['industry'] = fg[0]
```

In [ ]:
```
fg = fg.drop([0],axis=1)
```

In [ ]:
```
fg
```

Out[28]:

	industry
0	academia
1	academic_advising
2	academic_advisor
3	academic_websites
4	academics
...	...
1060	xpress_video_productions
1061	zoo
1062	zoo_keeping
1063	zoo_leisure_park
1064	zoology_environmental_studies

1065 rows × 1 columns

In [ ]:
```

```

In [ ]:
```
# dense =y.todense()
y = y.todense()
denselist = y.tolist()
df = pd.DataFrame(denselist, columns=feature_names)
```

In [ ]:
```
```

In [ ]:
```
max(df)
```

Out[195]:
```
'zoology_environmental_studies'
```

In [ ]:
```
df['academic_advising'].min()
```

Out[200]:
```
0
```

In [ ]:
```
import pickle
with open('vectorizer.pkl', 'wb') as handle:
    pickle.dump(vectorizer, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

In [ ]:
```
# y = y.astype('float16')
```

In [ ]:
```
X.shape
```

Out[149]:
```
(23931, 2)
```

In [ ]:
```
# y = multilabel_y.toarray()
```

In [ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10, random_
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2
print("Number of points in train data: ",X_train.shape[0])
print("Number of points in validation data: ",X_val.shape[0])
print("Number of points in test data: ",X_test.shape[0])
```

```
Number of points in train data:  17229
Number of points in validation data:  4308
Number of points in test data:  2394
```

In [ ]:
```
import numpy as np
tokens = Tokenizer()
train_text=X_train.Complete
val_text=X_val.Complete
test_text=X_test.Complete

tokens.fit_on_texts(train_text)


sequences_train = tokens.texts_to_sequences(train_text)
sequences_val = tokens.texts_to_sequences(val_text)
sequences_test = tokens.texts_to_sequences(test_text)




padded_text_train = pad_sequences(sequences_train, maxlen=200, padding='post')
padded_text_val = pad_sequences(sequences_val, maxlen=200, padding='post')
padded_text_test = pad_sequences(sequences_test, maxlen=200, padding='post')


padded_text_train = np.array(padded_text_train , dtype = np.int16)
padded_text_val = np.array(padded_text_val , dtype = np.int16)
padded_text_test = np.array(padded_text_test, dtype = np.int16)
```

In [ ]:
```
print(padded_text_train)
print(type(padded_text_train))
```

```
[[ 503   91   14 ...    0    0    0]
 [1053   34  237 ...    0    0    0]
 [  10  655   86 ...    0    0    0]
 ...
 [  50 2047    4 ...    0    0    0]
 [  10   52   79 ...    0    0    0]
 [  20    3  157 ...    0    0    0]]
<class 'numpy.ndarray'>
```

In [ ]:
```
vocab_size_text = len(tokens.word_index) + 1
```

In [ ]:
```
# url = "https://nlp.stanford.edu/data/glove.twitter.27B.zip"

!wget https://nlp.stanford.edu/data/glove.twitter.27B.zip
!unzip glove.twitter.27B.zip
```

```
--2021-08-04 06:12:04--  https://nlp.stanford.edu/data/glove.twitter.27B.zip (h
ttps://nlp.stanford.edu/data/glove.twitter.27B.zip)
Resolving nlp.stanford.edu (nlp.stanford.edu)... 171.64.67.140
Connecting to nlp.stanford.edu (nlp.stanford.edu)|171.64.67.140|:443... connect
ed.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://downloads.cs.stanford.edu/nlp/data/glove.twitter.27B.zip (htt
p://downloads.cs.stanford.edu/nlp/data/glove.twitter.27B.zip) [following]
--2021-08-04 06:12:05--  http://downloads.cs.stanford.edu/nlp/data/glove.twitte
r.27B.zip (http://downloads.cs.stanford.edu/nlp/data/glove.twitter.27B.zip)
Resolving downloads.cs.stanford.edu (downloads.cs.stanford.edu)... 171.64.64.22
Connecting to downloads.cs.stanford.edu (downloads.cs.stanford.edu)|171.64.64.2
2|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1520408563 (1.4G) [application/zip]
Saving to: ‘glove.twitter.27B.zip’

glove.twitter.27B.z 100%[===================>]   1.42G  4.81MB/s    in 4m 55s

2021-08-04 06:16:59 (4.92 MB/s) - ‘glove.twitter.27B.zip’ saved [1520408563/152
0408563]

Archive:  glove.twitter.27B.zip
  inflating: glove.twitter.27B.25d.txt
  inflating: glove.twitter.27B.50d.txt
  inflating: glove.twitter.27B.100d.txt
  inflating: glove.twitter.27B.200d.txt
```

In [ ]:
```
import pickle
import tqdm
embeddings_index = dict()

embeddings_index = dict()
f = open('glove.twitter.27B.200d.txt',encoding="utf8")
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

embedding_matrix = np.zeros((vocab_size_text, 200))
for word, i in tokens.word_index.items():
  embedding_vector = embeddings_index.get(word)
  if embedding_vector is not None:
    embedding_matrix[i] = embedding_vector

         embedding_matrix[i] = embedding_vector

print(len(embedding_matrix[0]))
```

```
200
```

In [ ]:
```
X_train.tags_tag_name.values[0]
```

Out[40]:
```
'art careerchoice careercounseling careerpath counseling governmentadministrati
on humanresources individualfamilyservices nonprofit retail socialwork youth yo
uthemployment'
```

In [ ]:
```

tags_train=X_train.tags_tag_name.values
tags_val=X_val.tags_tag_name.values
```

In [ ]:
```
tags_test=(X_test.tags_tag_name.values)
tokens = Tokenizer()
tokens.fit_on_texts(tags_train)
sequences_tags_train = tokens.texts_to_sequences(tags_train)
sequences_tags_val = tokens.texts_to_sequences(tags_val)
sequences_tags_test = tokens.texts_to_sequences(tags_test)
size_tags = len(tokens.word_index) + 1
```

In [ ]:
```
tokens = Tokenizer()
tokens.fit_on_texts(Data.professionals_industry)
sequences = tokens.texts_to_sequences(Data.professionals_industry)
```

In [ ]:
```
Data.professionals_industry
```

Out[122]:
```
0                      career_counseling computer_software
1                      insurance_law professional_training
2                                                 research
3                     professional_trainer writing_editing
4                             financial_services_education
                                   ...
23926
23927                                             zoo_keeping
23928    human_resources information_technology_service...
23929                                  education_management
23930            accounting_education education_management
Name: professionals_industry, Length: 23931, dtype: object
```

In [ ]:
```
tags_train
```

Out[43]:
```
array(['art careerchoice careercounseling careerpath counseling governmentadmin
istration humanresources individualfamilyservices nonprofit retail socialwork y
outh youthemployment',
       'civillitigation civilrights jobsearch justice law lawenforcement legal
litigation litigationsupport mediation police resume resumewriting socialimpact
socialwork sociology',
       'art careerchoice careercounseling careerpath computerscience computerso
ftware counseling gamedesign google graphicdesign humanresources individualfami
lyservices linux motiongraphics perl programming retail socialwork softwareengi
neering videogamedesign webdesign webdevelopment',
       ...,
       'alcoholsubstanceabuse budget business businessdevelopment college colle
geadvice collegemajor finance insurance internships interviewingskills intervie
ws jobsearch life managementconsulting relationshipmanagement resume resumewrit
ing sales',
       'bilingualspanish businessanalysis businessoperations businessprocess ch
emicals college costestimates engineering entertainment entrepreneurship intern
ational leadership management nonprofit overseas processimprovement projectmana
gement projectplanning projectteams projectdesign teambuilding teammanagement t
elecommunications volunteering worklifebalance',
       ''], dtype=object)
```

In [ ]:
```
c=[len(i) for i in sequences_tags_train]
np.percentile(c, 90)
```

Out[44]:
```
35.0
```

In [ ]:
```
padded_tags_train = pad_sequences(sequences_tags_train, maxlen=50,padding='post')
padded_tags_val = pad_sequences(sequences_tags_val, maxlen=50, padding='post')
padded_tags_test = pad_sequences(sequences_tags_test, maxlen=50, padding='post')

padded_tags_train = np.array(padded_tags_train , dtype = np.int16)
padded_tags_val = np.array(padded_tags_val , dtype = np.int16)
padded_tags_test = np.array(padded_tags_test , dtype = np.int16)
```

In [ ]:
```
size_tags
```

Out[46]: 6286

In [ ]:
```
# tensorflow.keras.backend.clear_session()

total_text_input_layer = Input(shape=(200), name = "total_sequence")
embedding_layer_total_text = Embedding(input_dim=vocab_size_text, output_dim=200,
lstm_total_text = LSTM(16, activation="relu", return_sequences=True)(embedding_la
flatten_total_text = Flatten()(lstm_total_text)



Layer = Dense(256, activation='relu', kernel_initializer='he_normal', name='dense
Layer = Dense(64, activation='relu', kernel_initializer='he_normal', name='dense_
Layer = Dense(32, activation='relu', kernel_initializer='he_normal', name='dense_
Layer = Dense(16, activation='relu', kernel_initializer='he_normal', name='dense_
output_layers = Dense(1065, activation='sigmoid', name='output_layers')(Layer)
model = Model(inputs=[total_text_input_layer], outputs=[output_layers])
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
total_sequence (InputLayer)  [(None, 200)]             0
_________________________________________________________________
embedding (Embedding)        (None, 200, 200)          3042000
_________________________________________________________________
lstm (LSTM)                  (None, 200, 16)           13888
_________________________________________________________________
flatten (Flatten)            (None, 3200)              0
_________________________________________________________________
dense_layer1 (Dense)         (None, 256)               819456
_________________________________________________________________
dense_layer2 (Dense)         (None, 64)                16448
_________________________________________________________________
dense_layer3 (Dense)         (None, 32)                2080
_________________________________________________________________
dense_layer4 (Dense)         (None, 16)                528
_________________________________________________________________
output_layers (Dense)        (None, 1065)              18105
=================================================================
Total params: 3,912,505
Trainable params: 870,505
Non-trainable params: 3,042,000
_________________________________________________________________
```

In [ ]:
```
from tensorflow.python.keras.callbacks import TensorBoard
from tensorflow.keras.callbacks import ModelCheckpoint
from time import time
```

In [ ]:
```
tensorboard = TensorBoard(log_dir="logs/{}".format(time))
filepath="weight_best.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_f1', verbose=1, save_best_onl
```

In [ ]:
```
import gc
gc.collect()
```

Out[51]:
```
1718
```

In [ ]:
```
import numpy as np
from tensorflow.keras.callbacks import Callback
from sklearn.metrics import confusion_matrix, f1_score, precision_score, recall_s
def f1(y_true,y_pred):
  return f1_score(y_true,y_pred,average = "micro")

# def micro_f1(y_true,y_prob):
#   y_pred = tf.math.argmax(y_prob,axis=1)
#   print("1,", y_pred)
#   print("1,", y_true)
#   return tf.py_function(f1,(y_true,y_pred),tf.double)
```

In [ ]:
```
y_val = np.array(y_val , dtype = np.int16)
```

In [ ]:
```
y_train = np.array(y_train , dtype= np.int16)
```

In [ ]:
```
!pip install tensorflow_addons
import tensorflow_addons as tfa
```

```
Collecting tensorflow_addons
  Downloading tensorflow_addons-0.13.0-cp37-cp37m-manylinux2010_x86_64.whl (679
kB)
     |████████████████████████████████| 679 kB 5.4 MB/s eta 0:00:01
Requirement already satisfied: typeguard>=2.7 in /usr/local/lib/python3.7/dist-
packages (from tensorflow_addons) (2.7.1)
Installing collected packages: tensorflow-addons
Successfully installed tensorflow-addons-0.13.0
```

In [ ]:
```
print(type(padded_text_train))
print(type(y_train))
print(type(padded_text_val))
print(type(y_val))
```

```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

In [ ]:
```
model.compile(optimizer='adam', loss='binary_crossentropy',metrics='accuracy')

model.fit(padded_text_train,
    y_train,
    batch_size=50,
    epochs=500,validation_data=(padded_text_val,y_val))
```

```
Epoch 1/500
345/345 [==============================] - 37s 101ms/step - loss: 0.0446 - ac
curacy: 0.0428 - val_loss: 0.0083 - val_accuracy: 0.0457
Epoch 2/500
345/345 [==============================] - 35s 100ms/step - loss: 0.0082 - ac
curacy: 0.0508 - val_loss: 0.0082 - val_accuracy: 0.0520
Epoch 3/500
345/345 [==============================] - 34s 100ms/step - loss: 0.0081 - ac
curacy: 0.0525 - val_loss: 0.0082 - val_accuracy: 0.0309
Epoch 4/500
345/345 [==============================] - 35s 102ms/step - loss: 0.0081 - ac
curacy: 0.0544 - val_loss: 0.0082 - val_accuracy: 0.0529
Epoch 5/500
345/345 [==============================] - 35s 100ms/step - loss: 0.0081 - ac
curacy: 0.0553 - val_loss: 0.0082 - val_accuracy: 0.0543
Epoch 6/500
345/345 [==============================] - 35s 103ms/step - loss: 0.0080 - ac
curacy: 0.0569 - val_loss: 0.0081 - val_accuracy: 0.0525
Epoch 7/500
```

In [ ]:
```
!pip install h5py
```

```
Requirement already satisfied: h5py in /usr/local/lib/python3.7/dist-packages
(3.1.0)
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist
-packages (from h5py) (1.5.2)
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-p
ackages (from h5py) (1.19.5)
```

In [ ]:
```
model.save('case_study_1_M3.hdf5')
```

In [ ]:
```
downloaded = drive.CreateFile({'id': '1rEXLUijrOxF3eKJXJSv2jx2SgjOR0mLQ'})
downloaded.GetContentFile('case_study_1_M3.hdf5')
```

In [ ]:
```
pd.set_option("display.max_colwidth", -1)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning:
Passing a negative integer is deprecated in version 1.0 and will not be support
ed in future version. Instead, use None to not limit the column width.
  """Entry point for launching an IPython kernel.
```

In [ ]:
```
Data.tags_tag_name.head(10)
```

Out[26]:
```
0    cybersecurity,disciplined,none,tech,womentech
1    business,career,careerpath,college,collegeadmissions,coverletters,governme
nt,insurance,internships,interviews,jobcoaching,law,major,minor,networking,prof
essionaltraining,psychology,psychometrics,resumewriting,school
2    brazil,children,coaching,distanceeducation,ngo,workingchildren
3    editing,instructionaldesign,professionaltraining,teaching,writing
4    education,educator,financialservices,generalinsurance,healthinsurance,insu
rance,math,mathematics,physics
5
6    agilemethodology,budgets,coaching,computergames,computergraphics,computin
g,crowdfunding,engineering,gamedesign,gamedevelopment,games,gaming,graphics,lea
dership,management,mobileapplications,mobiledevices,mobilegames,monetization,mo
torcycles,motorcycling,operations,organizationaldevelopment,podcasts,profession
aldevelopment,programmanagement,projectmanagement,projectplanning,qualityassura
nce,softwaredevelopment,softwareengineering,tabletopgames,talentmanagement,team
leadership,userinterfacedesign,videogames
7    advertising,agilemethodology,ajax,analytics,angularjs,ant,apidevelopment,a
utomation,bash,blogging,c,cassandra,centos,coffeescript,computer,computergames,
computerprogramming,computerscience,computersoftware,copywriting,css,databases,
design,development,distributedsystems,economics,editing,engine,enterprise,enter
prisesoftware,facebook,fiction,film,functionalprogramming,gamedesign,gamedevelo
pment,gameprogramming,gameplay,games,gaming,git,google,hadoop,hibernate,html,hu
manities,improvacting,java,javascript,jms,jquery,junit,language,languages,leade
rship,leadershiptraining,linux,literature,management,marketingstrategy,maven,mm
o,multithreading,mysql,networking,objectorienteddesign,oracle,perl,philosophy,p
hotography,php,productdevelopment,programmer,programming,projectmanagement,pyth
on,responsivedesign,rest,ruby,rubyrails,scala,scriptwriting,sdlc,search,sem,se
o,socialmedia,socialmediamarketing,socialnetworking,software,softwaredesign,sof
twaredevelopment,softwareengineering,stories,storytelling,subversion,sustainabi
lity,testdrivendevelopment,tomcat,traveling,tutoring,ubuntu,userexperiencedesig
n,userresearch,videogames,watertechnology,webdesign,webdevelopment,wordpress,wr
iting
8    clinicalpsychology,mentalhealth
9    architecture,design,interiordesign
Name: tags_tag_name, dtype: object
```

In [ ]:
```
Data.questions_body.head(10)
```

Out[33]:
```
0    junior h right thinking double major tech academy guaranteed internship th
inking worth two tech classes back back whole senior year take college classes
nearest community college know fact not soo much time give something not gonna
help worth information technology majoring technology internship time high scho
ol
1    currently undergrad want go law school lawyer thinking minoring psychology
would helpful long run psychology law
2    degree next biology marine
3    hi currently debating whether pursue career exercise sciences become gym t
eacher however upon researching job say demand gym teachers vanishing due chang
e county curriculum apparently gym class not even part county curriculum resour
ces say demand increased ever increase obese children need guidance make decisi
on major end not able pursue career would like please help teaching school exer
cise exercise science
4    hi high school senior set becoming pure mathematician ever since learned p
roof quadratic formula grade chasing abstract mathematics working improve probl
em solving skills less year ago came across interesting problem yet solve many
teachers ask help stumped leave discovery process worked problem months making
no significant progress decided stop therefore ask motivate continue know probl
em lead nowhere much time devoting problem math puremathematics
5    watch lot tv favorite show criminal minds watching show made interested st
udying forensic science college criminal justice forensic
6    hate college think waste time money work places without going college gami
ng videogamesmaking videogames
7    currently using sublime text although xcode used edit javascript html css
obj c etc files use really slick customizable companies require use ides ide sp
ecific reason not ide use specific personal programming related reason use ide
thank programming
8    would like become psychiatrist work people brain also medical field planni
ng majoring psychology wondering major pre med neurobiology anything like pre m
ed clinical psychology psychiatry psychiatrists
9    hello name patricia live charlotte north carolina wondering get job interi
or design got get future job interior design interior design interior decorator
Name: questions_body, dtype: object
```

In [ ]:
```
Data.questions_title.head(10)
```

Out[34]:
```
0    double major tech academy high school worth
1    declare minor undergrad want lawyer
2    get job prefered field
3    demand gym teachers diminishing
4    aspiring mathematician stay motivated researching pure mathematics
5    top colleges forensic science
6    take lot work college time work big companies like apple microsoft sony
anything video game related
7    programmers ide use
8    majors minors pursue want become psychiatrist
9    get job interior design
Name: questions_title, dtype: object
```

In [6]:
```
def function(question_body,question_title, tags):
  import re
  from keras.models import load_model
  import tensorflow
  import keras
  import pickle as pkl

  stopwords= set(['br', 'the', 'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'o
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itse
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'th
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than',
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "di
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma',
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn'
            'won', "won't", 'wouldn', "wouldn't"])
  from bs4 import BeautifulSoup
  from tqdm import tqdm
  preprocessed_questions_body = []
  preprocessed_questions_title = []
  preprocessed_tags = []

  def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    phrase = ' '.join(e.lower() for e in phrase.split() if e.lower() not in stopw
    # print(type(phrase))
    # print(phrase)
    return phrase
  preprocessed_questions_body = decontracted(question_body)
  preprocessed_questions_title= decontracted(question_title)
  preprocessed_tags = decontracted(tags)


  Complete = preprocessed_questions_body + " " + preprocessed_questions_title + '
  # print(Complete)
  with open('tokenizer.pkl', 'rb') as handle:
    tokenizeer=pkl.load(handle)
    sequences_test = tokenizeer.texts_to_sequences([Complete])
```

```
    padded_text_test = pad_sequences(sequences_test, maxlen=200, padding='post')
    print(padded_text_test.shape)

  model = load_model('case_study_1_M3.hdf5')
  # f_names = model.feature_name
  # model.feature_names = list(fg.industry.values)
  # f_names = model.feature_names
  # print(f_names)
  predicted_industry=model.predict(padded_text_test)
  # print(predicted_industry)

  with open('vectorizer.pkl', 'rb') as handle:
    vectorizer=pkl.load(handle)
  feature_names = vectorizer.get_feature_names()
  feature_names = pd.DataFrame(feature_names)

  thresh = 0.01 #0.01
  for dat in predicted_industry:
    whr = np.where(dat > thresh)

    # print(whr)


  predicted_class = feature_names.loc[whr]
  print("Predicted classes are :", predicted_class.values)


  return predicted_class
```

In [7]:
```
industry = function(question_body="i am student",question_title="i am good in bio
```

```
(1, 200)
Predicted classes are : [['education']
 ['entertainment']
 ['information_technology_services']
 ['management_consulting']
 ['telecommunications']]
```