In [1]:
```
from google.colab import drive
drive.mount('/drive/')
import os
os.chdir('/drive/My Drive/SKRA/NLP')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_i
d=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redi
rect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&scope=email%20https%3A%2F%2Fwww.go
ogleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdri
ve%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%20https%3
A%2F%2Fwww.googleapis.com%2Fauth%2Fpeopleapi.readonly&response_type=code (http
s://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pf
ee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%
3A2.0%3Aoob&scope=email%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdocs.test%2
0https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive%20https%3A%2F%2Fwww.googleapi
s.com%2Fauth%2Fdrive.photos.readonly%20https%3A%2F%2Fwww.googleapis.com%2Fauth%
2Fpeopleapi.readonly&response_type=code)

Enter your authorization code:
··········
Mounted at /drive/
```

In [0]:
```
import os
os.chdir('/drive/My Drive/SKRA/NLP')
```

In [4]:
```
ls
```

```
chatbot-countvectorizer-cosine.ipynb  chat_bot.csv  TextRanking.ipynb
```

In [11]:

```
import numpy as np
import pandas as pd
import nltk
from nltk import word_tokenize, sent_tokenize # tokenization
from nltk.stem import WordNetLemmatizer # Lemmatization
from nltk import pos_tag # pos tagging
#from nltk.stem import PorterStemmer # stemming
from nltk.corpus import wordnet # wordnet
import re # regular expression
from nltk.corpus import stopwords
stop = stopwords.words('english')
stop.remove('what')
stop.remove('which')
print(stop)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
"you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "i
t's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'who',
'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was',
'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'd
id', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'unt
il', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between',
'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from',
'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 't
hen', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'bo
th', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'no
t', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'wil
l', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o',
're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "did
n't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "have
n't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn',
"needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren',
"weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

In [12]:

```
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[12]:

```
True
```

In [14]:

```
chatbot = pd.read_csv('chat_bot.csv',encoding='latin-1')
chatbot.head()
```

Out[14]:

	Questions	Answers
0	What are the prerequisites for this Hadoop Tra...	There are no prerequisites for learning this c...
1	Do I need to know anything before leaning the ...	There are no prerequisites for learning this c...
2	Do I need to have some programming knowledge t...	There are no prerequisites for learning this c...
3	Is it mandatory to know some kind of programmi...	There are no prerequisites for learning this c...
4	Is programming important to learn Hadoop?	There are no prerequisites for learning this c...

In [0]:

```
def postag(pos):
  if pos.startswith('N'):
      wp = wordnet.NOUN
  elif pos.startswith('V'):
    wp = wordnet.VERB
  elif pos.startswith('R'):
    wp = wordnet.ADV
  elif pos.startswith('J'):
    wp = wordnet.ADJ
  else:
    wp = wordnet.NOUN

  return wp

wnl = WordNetLemmatizer() # intilize wordnetlemmatizer

def texprocess(doc):

  # step-1: lower the text
  doc = doc.lower()
  # step-2: remove special characters
  doc = re.sub(r'[^a-z]',' ',doc)
  # step-3: pos tagging (parts of speech)
  token = word_tokenize(doc) # tokenization - get the words
  token_pos = pos_tag(token) # tagging parts of speech
  # step-4: stemming
  #ps = PorterStemmer()
  #stemming = [ps.stem(word) for word in token]
  # step-4 : lemma and remove stopwords
  lemma = [wnl.lemmatize(word,pos=postag(pos)) for word,pos in token_pos if word

  clean = " ".join(lemma)
  return clean

def cosine(a,b):
  moda = np.linalg.norm(a) # magnitude of a
  modb = np.linalg.norm(b) # magnitude of b
  dotprod = np.dot(a,b) # dot product of vector a and vector b
  # a[0] , b[0] -> remove shape in it , we don't want vector to have some shape
  # i.e, neither column matrix nor row matrix
  cos = dotprod/(moda*modb)
  # print('INFO: similarity between document a and b is =',cos_theta)
  return cos
```

Word Embedding

- Count Vectorizer

Ranking Documents

- cosine similarity

$$cos(a,b) = \frac{\bar{a}.\bar{b}}{|a||b|}$$

In [0]:

```
documents = list(chatbot['Questions'])
# Step-1: Text processing
documents = [texprocess(doc) for doc in documents] # text processing of the all t
```

Step-2 : Word Embedding

In [21]:

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()

X = cv.fit_transform(documents).toarray() # word embedding count vectorizer
print('INFO: shape of array =',X.shape)
print('INFO: Features list =',cv.get_feature_names())
print('INFO: length of features =',len(cv.get_feature_names()))
```

```
INFO: shape of array = (726, 484)
INFO: Features list = ['able', 'accept', 'access', 'accredit', 'achive', 'acron
ym', 'actually', 'advantage', 'afternoon', 'agile', 'agility', 'ai', 'algorith
m', 'amason', 'amazon', 'analysis', 'analyst', 'analytics', 'anything', 'anywah
re', 'apace', 'apache', 'application', 'apply', 'approach', 'approch', 'archite
ch', 'architect', 'article', 'artificial', 'assistance', 'associate', 'attend',
'automation', 'available', 'average', 'aws', 'back', 'background', 'backup', 'b
ecome', 'behind', 'benefit', 'benificear', 'benifits', 'best', 'big', 'bigdat
a', 'blog', 'blue', 'body', 'bot', 'branch', 'break', 'buesness', 'build', 'bui
lding', 'bulk', 'bye', 'call', 'cancel', 'candidate', 'capstone', 'card', 'car
e', 'career', 'case', 'cd', 'certifaction', 'certificate', 'certification', 'ce
rtified', 'certify', 'chalenges', 'challenge', 'chef', 'ci', 'ciao', 'class',
'classification', 'classroom', 'cleaning', 'cloud', 'cod', 'come', 'common', 'c
ompany', 'complete', 'component', 'compponents', 'comprise', 'compute', 'comput
er', 'concept', 'conceptual', 'conduct', 'connect', 'consider', 'contact', 'con
tent', 'continue', 'continuo', 'continuous', 'cost', 'coureses', 'course', 'cou
rsework', 'cover', 'credit', 'dat', 'data', 'datasets', 'day', 'degree', 'deliv
ery', 'demand', 'demo', 'depend', 'deployment', 'desirable', 'developer', 'deve
lopment', 'device', 'devopa', 'devops', 'devovps', 'devsecops', 'difference',
'different', 'differentiate', 'differnt', 'difficult', 'discount', 'docker', 'd
omains', 'dude', 'dvoups', 'earn', 'economics', 'effective', 'effort', 'eligibi
lity', 'employer', 'engineer', 'enrol', 'enroll', 'enrollment', 'entail', 'envi
ronment', 'etc', 'even', 'evening', 'everyone', 'exam', 'example', 'expect', 'e
xpectation', 'experience', 'explain', 'express', 'extention', 'extremely', 'fac
e', 'factor', 'faculti', 'faculty', 'fail', 'fee', 'field', 'find', 'finish',
'flume', 'follow', 'form', 'framework', 'free', 'fresher', 'future', 'get', 'gi
ve', 'global', 'go', 'good', 'guarantee', 'guidance', 'hadoop', 'hand', 'happen
ing', 'hear', 'heard', 'hello', 'help', 'helpful', 'hey', 'heyyo', 'hi', 'hir
e', 'history', 'hit', 'hive', 'hope', 'hot', 'hour', 'hub', 'implement', 'imple
mentation', 'implemetation', 'implrmentation', 'important', 'improve', 'includ
e', 'increase', 'independent', 'india', 'indusry', 'industry', 'innomatics', 'i
nstitute', 'institution', 'integration', 'intelligence', 'interview', 'involv
e', 'issue', 'jenkins', 'job', 'join', 'key', 'kind', 'know', 'knowledge', 'la
b', 'language', 'laptop', 'lean', 'learn', 'learning', 'leave', 'less', 'licens
e', 'like', 'listen', 'little', 'live', 'locate', 'location', 'log', 'long', 'l
ook', 'machine', 'macro', 'macros', 'main', 'makeup', 'management', 'mandator
y', 'many', 'mapreduce', 'market', 'material', 'math', 'mathematics', 'matter',
'mean', 'median', 'mention', 'methodology', 'mine', 'mining', 'miss', 'ml', 'mo
de', 'model', 'money', 'morning', 'much', 'must', 'name', 'near', 'necessary',
'need', 'new', 'next', 'night', 'objective', 'offer', 'office', 'one', 'onlin
e', 'open', 'opperations', 'option', 'organisation', 'organization', 'others',
'overall', 'part', 'pas', 'pass', 'past', 'path', 'pay', 'payment', 'payslip',
'pega', 'perfect', 'period', 'person', 'personal', 'perspective', 'pipeline',
'place', 'placement', 'plan', 'platform', 'podcasts', 'policy', 'popular', 'pos
se', 'possible', 'post', 'powerful', 'practice', 'pre', 'prediction', 'preferre
d', 'prepare', 'prepping', 'prerecord', 'prerequisite', 'present', 'price', 'pr
ior', 'priority', 'prism', 'problem', 'process', 'product', 'profession', 'prof
essional', 'program', 'programming', 'project', 'prolific', 'proper', 'prospe
r', 'provide', 'purpose', 'put', 'python', 'qualifications', 'rdm', 'real', 're
ally', 'receive', 'recommend', 'recommended', 'record', 'recruit', 'reduce', 'r
eduction', 'reexamination', 'reference', 'refund', 'register', 'relate', 'remot
e', 'replace', 'require', 'requirement', 'result', 'resume', 'retake', 'retur
n', 'robotic', 'role', 'rpa', 'run', 'salary', 'scala', 'schedule', 'science',
'scientist', 'scope', 'script', 'scripting', 'scrum', 'security', 'see', 'selec
t', 'selenium', 'service', 'session', 'set', 'significance', 'similar', 'simpli
learn', 'skill', 'skills', 'solution', 'soon', 'source', 'spark', 'spend', 'sq
l', 'stage', 'stand', 'start', 'statistic', 'step', 'store', 'study', 'succee
d', 'successful', 'suggest', 'suitable', 'sup', 'support', 'sure', 'syllabus',
'system', 'ta', 'take', 'taught', 'teach', 'teacher', 'teaching', 'team', 'tech
nical', 'technique', 'technologies', 'technology', 'tell', 'testimonial', 'thin
g', 'think', 'time', 'tipical', 'today', 'tool', 'top', 'topic', 'train', 'trai
ner', 'training', 'transformation', 'type', 'typical', 'ui', 'uipath', 'unloc
k', 'us', 'use', 'used', 'useful', 'usefull', 'user', 'usually', 'valid', 'vali
dation', 'valuable', 'video', 'waht', 'waive', 'waiver', 'want', 'watch', 'wate
rfall', 'way', 'wazzup', 'web', 'week', 'well', 'wep', 'what', 'whats', 'whic
h', 'without', 'wonderful', 'work', 'world', 'would', 'ya', 'yes']
INFO: length of features = 484
```

Finding Similar documents

In [0]:

```python
import operator
```

In [0]:

```python
def chatanswers(query):

  # step-1: text processing
  clean = texprocess(query)
  # step-2: word embedding (count vectorizer)
  b = cv.transform([query]).toarray() # query in list

  cosvalue ={}
  for i,vector in enumerate(X):
    cos = cosine(vector,b[0]) # b[0] -> remove shape in it
    cosvalue.update({i:cos}) # append values in dictonary

  #df['cos'] = cosvalue.values()
  #df.sort_values(by='cos',ascending=False)
  sort = sorted(cosvalue.items(), key=operator.itemgetter(1),reverse=True)
  ind = [index for index,cosv in sort[:5]][0]
  return ind,str(chatbot.loc[ind]['Answers'])
```

In [47]:

```python
query = 'what is data science ?'
index, ans = chatanswers(query)
print(ans)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:41: RuntimeWarnin
g: invalid value encountered in true_divide
```

In [51]:

```
while True:

  chatinput = input('Srikanth: ')
  if chatinput == 'exit':
    print('Thank you very much have a nice day !!!')
    break

  ind, ans = chatanswers(chatinput)
  print(ans)
```

```
Srikanth: exit
Thank you very much have a nice day !!!
```

In [0]:

```
```