Steps followed to achieve the result in assignment 2:

Question: Write a code to find at least 20 frequently asked questions for the subjects and topics mentioned in the dataset ([http://bit.ly/epo\_FAQ\_sample\_topics](https://www.google.com/url?q=http://bit.ly/epo_FAQ_sample_topics&sa=D&ust=1544962804005000&usg=AFQjCNHxifZ0w_haVJcXLTNnyC262o9TSQ)) from the questions in the dataset ([http://bit.ly/epo\_FAQ\_superset\_Sample](https://www.google.com/url?q=http://bit.ly/epo_FAQ_superset_Sample&sa=D&ust=1544962804005000&usg=AFQjCNGRv0GV_hVZQuWDFFj8Bqyq8cu_dQ)). The second dataset contains list of 500 questions that was asked in the interview.

**Solution:** I built my solution using python. Libraries required are Numpy, pandas, nltk.

Steps followed:

1. Read the question dataset from csv file.
2. Split the dataset in questions having one or more sentences.
3. Made the corpora for each subject given in the topic dataset so that based on the keywords in corpora we can classify each question into its subject and their respective topics.
4. Tokenize each question into words.
5. Remove the words which are general in use also known as stop words. Also remove the question words.
6. Lemmatize each token means deriving the base word like base word of “better” is “good”.
7. Based on the tokens present in the question, classify them in their topics and subject by matching them with corpora. In this way, every question will be classified according to its topic and subject.

For matching words, I match by substring matching to avoid difference between “inherit”, ”inheritance”, “inherited” etc.

1. After splitting each question into their keywords. Match each keyword with the keywords of other question and based on this, assign ranking to each question. Higher rank means its keywords have matched with more other questions.
2. After ranking, iterate each question in decreasing order of rank and check their subject assigned. For each subject, choose top 15 of them.

In this way, Most frequently asked questions will be sorted out subject wise. I solved this problem in two data set files.

On file include question in overall frequency wise.

Link to first file: <https://drive.google.com/open?id=1J9iI_NXtMZM14KvxVWGsjIkhJm6L8xHT>

Second file is in which question are arranged in subject wise manner having more frequency at the top.

Link to second file: <https://drive.google.com/open?id=1umgncwDe42s74gl7acbgiVUoXjyfdBDZ>

**Python Code:**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

def match(w1,w2):

c=0

x=min(len(w1),len(w2))

for i in range(x):

if w1[i]==w2[i]:

c+=1

if x<4 and c==x:

return True

elif x>=4 and c/x>0.7:

return True

else:

return False

fo=open("500\_questions.csv","r")

data=fo.read()

data=data.split("\n")

data=data[1:-1]

rank={}

for i in range(len(data)):

rank[i]=0

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

stop\_words = set(stopwords.words('english'))

from collections import defaultdict

top=defaultdict(lambda:('',''))

sub=['Operating System','Database Management System','Object\_oriented\_programming']

os=['fcfs','buffering','deadlock','caching','security','scheduling']

db=['dml','joints','ddl','transaction','keys','sql']

oo=['jdbc','applet','polymorphism','thread','inheritance','encapsulation','Object']

a0=[['fcfs','first serve','first come'],['buff','speed','i/o'],['deadlock'],['cach','pag','map'],['secur','authen','threat','password','protec'],

['schedul','algorithm','preemp']]

a1=[['dml','manipulation language','insert','alter'],['join'],['ddl','create','drop','constrain','integr'],['transact','serial','commit','rollback'],

['primary','superkey','foreign','key'],['sql','dbms']]

a2=[['jdbc','driver','connect'],['applet','webpage','html','tag'],['polymorph','overload','overrid'],['thread','multitask','runnable'],

['inherit','subclass','super','extend'],['encapsul','abstract'],['class','object','oop']]

d1=[]

que=['question','questions','how','when','what','which','why']

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

countsen=0

for sen in data:

word\_tokens = word\_tokenize(sen)

filtered\_sentence = []

f1=True

for w in word\_tokens:

#if w.isnumeric() or w.isalnum():

# continue

w=w.lower()

if w not in stop\_words and len(w)>2 and w not in que:

w=lemmatizer.lemmatize(w)

w=lemmatizer.lemmatize(w,pos='a')

w=lemmatizer.lemmatize(w,pos='v')

filtered\_sentence.append(w)

if f1:

for i in range(len(a0)):

for j in range(len(a0[i])):

if w.count(a0[i][j])>0:

top[countsen]=(os[i],sub[0])

f1=False

break

for i in range(len(a1)):

for j in range(len(a1[i])):

if w.count(a1[i][j])>0:

top[countsen]=(db[i],sub[1])

f1=False

break

for i in range(len(a2)):

for j in range(len(a2[i])):

if w.count(a2[i][j])>0:

top[countsen]=(oo[i],sub[2])

f1=False

break

d1.append(filtered\_sentence[1:])

countsen+=1

for i in range(len(d1)):

for j in range(len(d1)):

flag=False

for k in d1[i]:

for l in d1[j]:

if match(k,l):

rank[i]+=1

flag=True

break

if flag:

break

cdb=0

cos=0

coo=0

appendFile = open('filteredtext.csv','a')

appendFile.write(" Topic, Subject, Question,\n")

appendFile.close()

mainst="Topic, Subject, Question,\n"

osst=""

dbst=""

oost=""

for k,v in sorted(rank.items(),reverse=True, key =lambda kv:(kv[1], kv[0])):

if cdb>14 and coo>14 and cos>14:

break

flag=False

if top[k][1]==sub[0] and cos<=14:

senten=data[k].split(',',1)[1]

appendFile = open('filteredtext.csv','a')

appendFile.write(top[k][0]+", "+top[k][1]+", "+senten+",\n")

appendFile.close()

osst+=top[k][0]+", "+top[k][1]+", "+senten+",\n"

cos+=1

if top[k][1]==sub[1] and cdb<=14:

senten=data[k].split(',',1)[1]

appendFile = open('filteredtext.csv','a')

appendFile.write(top[k][0]+", "+top[k][1]+", "+senten+",\n")

appendFile.close()

dbst+=top[k][0]+", "+top[k][1]+", "+senten+",\n"

cdb+=1

if top[k][1]==sub[2] and coo<=14:

senten=data[k].split(',',1)[1]

appendFile = open('filteredtext.csv','a')

appendFile.write(top[k][0]+", "+top[k][1]+", "+senten+",\n")

appendFile.close()

oost+=top[k][0]+", "+top[k][1]+", "+senten+",\n"

coo+=1

newfile=open('result.csv','w')

newfile.write(mainst+oost+dbst+osst)

newfile.close()