_This Project contains 3 modules under the name "The_document_helper" namely :-

1) _Module : CV Assistant.

2) _Module : Pdf Merger.

3) _Module : Pdf to text Converter. ¶

.....

1) Module : CV Assistant

_Description : Analysing pdf and identifying the maximum occuring keywords , then using the keywords to check for curse words and giving sugestions to replace them

: (internet connectivity required)

: IT gives meaning and synonyms of the curse words ,suggestive words for replacing the curse words.

_Usage : Can be used as an invigilator to your CV OR RESUME (checking for curse words and identifying the frequency of keywords you used and then suggesting words against curse words)

STEP 1 - using tkinter open dialog for selecting your resume or any other document

In [1]:
```
import tkinter
from tkinter import *
from tkinter.filedialog import askopenfilename
from tkinter.messagebox import showerror

class MyFrame(Frame):
    def __init__(self):
        Frame.__init__(self)
        self.master.title("PDF ANALYSIS FOR CURSE WORDS")
        self.master.rowconfigure(5, weight=1)
        self.master.columnconfigure(5, weight=1)
        self.grid(sticky=W+E+N+S)

        self.button = Button(self, text="Choose  your pdf ", command=self.load_file, width=100)
        self.button.grid(row=1, column=0, sticky=W)

    def load_file(self):
        fname = tkinter.filedialog.askopenfilename(filetypes=(("Pdf files", "*.pdf")
                                        # ,("HTML files", "*.html;*.htm"),
                                         ,("All files", "*.*") ))#used for multiple files selection


        if fname:
            try:
                print("Success")
            except:                     # <- naked except is a bad idea
                showerror("Open Source File", "Failed to read file\n'%s'" % fname)
            return fname

a=MyFrame().load_file()
print(a)
```

```
Success
/home/arjun/13_april/project file and cv/updated cv/ARJUN RESUME CSe.pdf
```

STEP 2 - identifying frequency of keywords used `

In [2]:
```
import PyPDF2
import textract
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
#nltk.download('stopwords')

pdfFileObj = open(a,'rb')
#The pdfReader variable is a readable object that will be parsed
pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
#discerning the number of pages will allow us to parse through all #the pages
num_pages = pdfReader.numPages
count = 0
text = ""
#The while loop will read each page
while count < num_pages:
    pageObj = pdfReader.getPage(count)
    count +=1
    text += pageObj.extractText()
#This if statement exists to check if the above library returned #words. It's done because PyPDF2 cannot reac
if text != "":
   text = text
#If the above returns as False, we run the OCR library textract to #convert scanned/image based PDF files int
else:
   text = textract.process(fileurl, method='tesseract', language='eng')
# Now we have a text variable which contains all the text derived #from our PDF file. Type print(text) to see
# Now, we will clean our text variable, and return it as a list of keywords.

#The word_tokenize() function will break our text phrases into #individual words
tokens = word_tokenize(text)
#we'll create a new list which contains punctuation we wish to clean
punctuations = ['(',')',';',':','[',']',',']
#We initialize the stopwords variable which is a list of words like #"The", "I", "and", etc. that don't hold
stop_words = stopwords.words('english')
#We create a list comprehension which only returns a list of words #that are NOT IN stop_words and NOT IN pur
keywords = [word for word in tokens if not word in stop_words and  not word in punctuations]

frequency_list = {}
for word in keywords:
    if word not in frequency_list:
        frequency_list[word] = 1
```

```
    else:
        frequency_list[word] += 1
#print(frequency_list)
print([(k,v) for v,k in sorted([(v,k) for k,v in frequency_list.items()],reverse=True)])
```

```
[('.', 35), ('Title', 12), ('Description', 10), ('using', 5), ('http', 5), ('system', 4), ('online', 4), ('e
xamination', 4), ('developed', 4), ('calling', 4), ('Verbose', 4), ('Python', 4), ('Paper', 4), ('Organizati
on', 4), ('Guess', 4), ('Generator', 4), ('Duration', 4), ('Android', 4), ('application', 3), ('Word', 3),
('Lucknow', 3), ('//demo.mediatrenz.com/citystyle/', 3), ('-', 3), ('year', 2), ('work', 2), ('store', 2),
('shopping', 2), ('question', 2), ('probability', 2), ('previous', 2), ('press', 2), ('papers', 2), ('pape
r', 2), ('organization', 2), ('months', 2), ('messaging', 2), ('maximum', 2), ('knowledge', 2), ('generate
s', 2), ('free', 2), ('features', 2), ('facilities', 2), ('digipodium', 2), ('develope', 2), ('data', 2),
('cost', 2), ('chatting', 2), ('built', 2), ('best', 2), ('based', 2), ('automatically', 2), ('application
s', 2), ('app', 2), ('analyses', 2), ('Used', 2), ('URL', 2), ('Technical', 2), ('System', 2), ('Software',
2), ('Real-time', 2), ('Questions', 2), ('Projects', 2), ('Platform', 2), ('Online', 2), ('MySQL', 2), ('Mul
tiple', 2), ('MCQ', 2), ('Learned', 2), ('J2ee', 2), ('It', 2), ('Examination', 2), ('Engineer', 2), ('Dat
a', 2), ('College', 2), ('Chatting', 2), ('Calling', 2), ('AMCAT', 2), ('3', 2), ('%', 2), ('fiQuiz', 1), ('fi
According', 1), ('————..', 1), ('————', 1), ('ŒUtkarsh', 1), ('ŒIT', 1), ('working', 1), ('weeks', 1), ('w
ebsites', 1), ('website', 1), ('ven', 1), ('used', 1), ('um', 1), ('training', 1), ('tomcat', 1), ('thinkin
g', 1), ('srivastavarjun23', 1), ('solutions', 1), ('servlets', 1), ('server', 1), ('science', 1), ('role',
1), ('relations', 1), ('quick', 1), ('quest', 1), ('python', 1), ('profiles', 1), ('profess', 1), ('proactiv
e', 1), ('popular', 1), ('oriented', 1), ('offering', 1), ('month', 1), ('mining', 1), ('maxim', 1), ('lik
e', 1), ('learner', 1), ('learned', 1), ('latest', 1), ('knowl', 1), ('jsp', 1), ('java', 1), ('ipython',
1), ('ional', 1), ('interpersonal', 1), ('information', 1), ('ide', 1), ('hereby', 1), ('growth', 1), ('gro
w', 1), ('good', 1), ('gmail', 1), ('gle', 1), ('gives', 1), ('giv', 1), ('gi', 1), ('following', 1), ('fetc
hing', 1), ('excellence', 1), ('etc', 1), ('es', 1), ('employable', 1), ('edge', 1), ('ecommerce', 1), ('eWe
bGuru', 1), ('dynamic', 1), ('development', 1), ('develop', 1), ('details', 1), ('desktop', 1), ('declared',
1), ('days', 1), ('crawler', 1), ('correct', 1), ('communication', 1), ('com', 1), ('cloud', 1), ('cetpa',
1), ('ce', 1), ('cation', 1), ('business', 1), ('building', 1), ('avail', 1), ('attention', 1), ('appli',
1), ('android', 1), ('analyzing', 1), ('analytical', 1), ('analysis', 1), ('active', 1), ('achieve', 1), ('a
ble', 1), ('ability', 1), ('YEAR', 1), ('XII', 1), ('X', 1), ('Working', 1), ('Wordpress', 1), ('Web', 1),
('UNIVERSITY', 1), ('Trophy', 1), ('Training', 1), ('To', 1), ('This', 1), ('Tel', 1), ('Studio', 1), ('Stre
ngths', 1), ('Socialefl', 1), ('Services', 1), ('Score', 1), ('Science', 1), ('STANDARD', 1), ('SRIVASTAVA',
1), ('Result', 1), ('Res', 1), ('Qualification', 1), ('Project', 1), ('Product', 1), ('Press', 1), ('Playe
d', 1), ('Place', 1), ('Pillow', 1), ('Pandas', 1), ('PURSUING', 1), ('PERCENTAGE', 1), ('PASSING', 1), ('O
F', 1), ('OBJECTIVE', 1), ('Noida', 1), ('Mobile', 1), ('Jupiter', 1), ('J2EE', 1), ('Internship', 1), ('Inf
oTech', 1), ('Industrial', 1), ('IT', 1), ('ISCE', 1), ('ISC', 1), ('ID', 1), ('I', 1), ('Goo', 1), ('GUI',
1), ('Fundamentals', 1), ('Firebase', 1), ('Fest', 1), ('Event', 1), ('Email', 1), ('Eclipse', 1), ('ENGINEE
RING', 1), ('Declaration', 1), ('Date', 1), ('DAS', 1), ('Coordinator', 1), ('Choice', 1), ('Choi', 1), ('CS
E', 1), ('COLLEGE', 1), ('BOARD', 1), ('BANARASI', 1), ('BABU', 1), ('B.TECH', 1), ('Assistant', 1), ('Api',
```

```
1), ('Apache', 1), ('Also', 1), ('Advanced', 1), ('Achievements', 1), ('Academic', 1), ('ARJUN', 1), ('@',
1), ('83.42', 1), ('80.00', 1), ('4', 1), ('3.6', 1), ('3.0', 1), ('2018', 1), ('2017', 1), ('2014', 1), ('2
010', 1), ('15', 1), ('1', 1), ('//www.brazilianvirginhumanhair.com/', 1), ('//demo3.mediatrenz.com/Theparis
closet/', 1), ('/', 1), ('+91-9598917733', 1), ('+91-8090958285', 1)]
```

STEP 2 - Performing Profanity Check on identified frequency keywords used `

In [3]:
```
#import urllib.request
import requests
lis=[]

#below is a function for progress bar
def log_progress(sequence, every=None, size=None, name='Items'):
    from ipywidgets import IntProgress, HTML, VBox
    from IPython.display import display

    is_iterator = False
    if size is None:
        try:
            size = len(sequence)
        except TypeError:
            is_iterator = True
    if size is not None:
        if every is None:
            if size <= 200:
                every = 1
            else:
                every = int(size / 200)     # every 0.5%
    else:
        assert every is not None, 'sequence is iterator, set every'

    if is_iterator:
        progress = IntProgress(min=0, max=1, value=1)
        progress.bar_style = 'info'
    else:
        progress = IntProgress(min=0, max=size, value=0)
    label = HTML()
    box = VBox(children=[label, progress])
    display(box)

    index = 0
    try:
        for index, record in enumerate(sequence, 1):
            if index == 1 or index % every == 0:
                if is_iterator:
                    label.value = '{name}: {index} / ?'.format(
                        name=name,
                        index=index
```

```
                    )
                else:
                    progress.value = index
                    label.value = u'{name}: {index} / {size}'.format(
                        name=name,
                        index=index,
                        size=size
                    )
            yield record
    except:
        progress.bar_style = 'danger'
        raise
    else:
        progress.bar_style = 'success'
        progress.value = index
        label.value = "{name}: {index}".format(
            name=name,
            index=str(index or '?')
        )

#below is a function for finding the curse words
def check_content_profanity(contents):
    print(' Checking for Curse words , Please wait....\n')
    flag=0
    count=0
    for x in log_progress(contents,every=1):
        output = requests.get('http://www.wdylike.appspot.com/?q='+x).text
        #connection = urllib.request.urlopen('http://www.wdylike.appspot.com/?q='+x)
        #output=connection.read()
        #print(x+'  '+output)
        if(output=='true'):
            lis.append(x)
            count+=1
            print(x+"\n is a curse word ")
            flag=1
    if(flag==1):
            print('\n The document has ' + str(count) +' curse words and must be reconsidered for content ')
    else:
        print('\n The document has no curse words and is safe for usage ')
    #connection.close()

check_content_profanity(frequency_list.keys())
```

```
RequestsDependencyWarning: urllib3 (1.22) or chardet (2.3.0) doesn't match a supported version! [__init__.p
y:80]

 Checking for Curse words , Please wait....
```

Items: 265

```
PASSING
 is a curse word
Assistant
 is a curse word

 The document has 2 curse words and must be reconsidered for content
```

STEP 3 - Finding meaning and synonyms of the identified curse words `

In [4]:
```
from nltk.corpus import wordnet

def find_synonym(x):
    synonyms = []
    for syn in wordnet.synsets(x):
        for lemma in syn.lemmas():
            synonyms.append(lemma.name())
    return (set(synonyms))

for x in lis:
    syn = wordnet.synsets(x)
    print('\n'+x+' MEANS \n'+syn[0].definition()+'\n')
    print('SYNONYMS of '+x+' are \n')
    print(find_synonym(x))
```

```
PASSING MEANS
(American football) a play that involves one player throwing the ball to a teammate

SYNONYMS of PASSING are

{'fall', 'kick_the_bucket', 'ephemeral', 'blow_over', 'give-up_the_ghost', 'go_across', 'reach', 'turn_ove
r', 'go_on', 'transcend', 'drop_dead', 'authorise', 'lead', 'fall_out', 'exceed', 'perish', 'qualifying', 'p
ass_off', 'pass', 'authorize', 'release', 'go', 'make_pass', 'overtake', 'return', 'passing_play', 'draw',
'clear', 'pass_along', 'passing', 'short-lived', 'evanesce', 'expiration', 'go_along', 'passing_game', 'casu
al', 'overstep', 'loss', 'sink', 'give', 'overhaul', 'come_about', 'snuff_it', 'legislate', 'go_by', 'pop_of
f', 'communicate', 'perfunctory', 'slip_by', 'fleet', 'put_across', 'cursory', 'run', 'guide', 'fade', 'make
_it', 'go_past', 'travel_by', 'buy_the_farm', 'lapse', 'going', 'egest', 'occur', 'expire', 'super', 'exit',
'surpass', 'eliminate', 'transitory', 'take_place', 'top', "cash_in_one's_chips", 'passage', 'happen', 'croa
k', 'go_through', 'pass_away', 'exceedingly', 'hand', 'elapse', 'overtaking', 'choke', 'decease', 'extend',
'extremely', 'pass_by', 'excrete', 'slip_away', 'spend', 'devolve', 'slide_by', 'hap', 'departure', 'fugacio
us', 'transient', 'die', 'conk', 'glide_by', 'pass_on'}

Assistant MEANS
a person who contributes to the fulfillment of a need or furtherance of an effort or purpose

SYNONYMS of Assistant are

{'help', 'supporter', 'helper', 'adjunct', 'assistant'}
```

STEP 4 - Finding suggestions to replace the identified curse words `

In [5]:
```
import urllib.request
def suggestion(words):
    lis1=[]
    #print(' Checking for Suggestions , Please wait....\n')
    for x in log_progress(words,every=1):
        connection = urllib.request.urlopen('http://www.wdylike.appspot.com/?q='+x)
        output=connection.read()
        if(output==b'false'):
            lis1.append(x)
    connection.close()
    print(lis1)


for x in lis:
    print('\n SUGGESTIONS TO REPLACE '+x+' are ...\n')
    y=find_synonym(x)
    suggestion(y)
```

```
 SUGGESTIONS TO REPLACE PASSING are ...
```

Items: 99

```
['fall', 'kick_the_bucket', 'ephemeral', 'blow_over', 'give-up_the_ghost', 'go_across', 'reach', 'turn_ove
r', 'go_on', 'transcend', 'drop_dead', 'authorise', 'lead', 'fall_out', 'exceed', 'perish', 'qualifying', 'a
uthorize', 'release', 'go', 'overtake', 'return', 'draw', 'clear', 'short-lived', 'evanesce', 'expiration',
'go_along', 'casual', 'overstep', 'loss', 'sink', 'give', 'overhaul', 'come_about', 'snuff_it', 'legislate',
'go_by', 'pop_off', 'communicate', 'perfunctory', 'slip_by', 'fleet', 'put_across', 'cursory', 'run', 'guid
e', 'fade', 'make_it', 'go_past', 'travel_by', 'buy_the_farm', 'lapse', 'going', 'egest', 'occur', 'expire',
'super', 'exit', 'eliminate', 'transitory', 'take_place', 'top', "cash_in_one's_chips", 'happen', 'croak',
'go_through', 'exceedingly', 'hand', 'elapse', 'overtaking', 'choke', 'decease', 'extend', 'extremely', 'exc
rete', 'slip_away', 'spend', 'devolve', 'slide_by', 'hap', 'departure', 'fugacious', 'transient', 'die', 'co
nk', 'glide_by']

 SUGGESTIONS TO REPLACE Assistant are ...
```

Items: 5

```
['help', 'supporter', 'helper', 'adjunct']
```

Restarting kernel after end of module

In [*]:
```
from IPython.core.display import HTML
HTML("<script>Jupyter.notebook.kernel.restart()</script>")
```

Out[6]:

2) Module : pdf merger

_Description : Selecting multiple pdf files with dialog and merging selected files into a single pdf named "combined*result.pdf*"

_Usage : for example - can be used to merge all exam papers and specific study material into a single common pdf

STEP 1 - defining tkinter open dialog for multiple selection and show selected files as a list

In [2]:
```
import tkinter
from tkinter import *
from tkinter.filedialog import askopenfilename
from tkinter.messagebox import showerror

class MyFrame(Frame):
    def __init__(self):
        Frame.__init__(self)
        self.master.title("PDF MERGER")
        self.master.rowconfigure(5, weight=1)
        self.master.columnconfigure(5, weight=1)
        self.grid(sticky=W+E+N+S)

        self.button = Button(self, text="Choose previous year pdf papers", command=self.load_file, width=100)
        self.button.grid(row=1, column=0, sticky=W)

    def load_file(self):
        fname = tkinter.filedialog.askopenfilenames(filetypes=(("Pdf files", "*.pdf")
                                        #,("HTML files", "*.html;*.htm"),
                                        ,("All files", "*.*") ))#used for multiple files selection
        root=tkinter.Tk()
        files=root.tk.splitlist(fname)
        pdf=list(files)

        if fname:
            try:
                print("Success")
            except:                     # <- naked except is a bad idea
                showerror("Open Source File", "Failed to read file\n'%s'" % fname)
            return pdf
```

STEP-2 CALLING tkinter open dialog and GETTING list from the class defined in the variable a , printing the files selected with their path , and also printing length of list i.e no. of files selected

In [3]:
```
a=MyFrame().load_file()
print(a)
print(len(a))
```

```
Success
['/home/arjun/13_april/3 march pic_paper/8th sem b.tech/NCS-080  Pattern Recognition/btech-cs-7-sem-pattern-
recognition-ecs-074-2016.pdf', '/home/arjun/13_april/3 march pic_paper/8th sem b.tech/NCS-080  Pattern Recog
nition/btech-cs-8-sem-pattern-recognition-ncs-080-2017.pdf', '/home/arjun/13_april/3 march pic_paper/8th sem
b.tech/NCS-080  Pattern Recognition/btech-it-7-sem-pattern-recognition-ecs-074-2016.pdf', '/home/arjun/13_ap
ril/3 march pic_paper/8th sem b.tech/NCS-080  Pattern Recognition/ecs-074 pattern recognition 2011-12.pdf',
'/home/arjun/13_april/3 march pic_paper/8th sem b.tech/NCS-080  Pattern Recognition/ecs-074 pattern recognit
ion 2012-13.pdf', '/home/arjun/13_april/3 march pic_paper/8th sem b.tech/NCS-080  Pattern Recognition/ecs-07
4 pattern recognition 2013-14.pdf', '/home/arjun/13_april/3 march pic_paper/8th sem b.tech/NCS-080  Pattern
Recognition/ecs-074 pattern recognition 2014-15.pdf', '/home/arjun/13_april/3 march pic_paper/8th sem b.tec
h/NCS-080  Pattern Recognition/ecs074 pattern recognition 2015-16.pdf']
8
```

STEP-3 LIST containing path of files selected stored in a variable b to access later on

In [4]:
```
b=a
print(type(b))
```

```
<class 'list'>
```

STEP-4 MERGING OF PDF and saving pdf in location of current notebook

In [5]:
```
from PyPDF2 import PdfFileMerger, PdfFileReader
merger = PdfFileMerger()
for filename in b:
    merger.append(PdfFileReader(filename, 'rb'))

merger.write("combined_result.pdf")
```

.....

Restarting kernel after end of module

In [*]:
```
from IPython.core.display import HTML
HTML("<script>Jupyter.notebook.kernel.restart()</script>")
```

Out[6]:

3) Module : pdf to text converter

_Description : Selecting pdf file with dialog and converting it to equivalent text file

_Usage :The text file created Can be used to perform easy data analysis on the desired pdf file

STEP 1 - using tkinter open dialog for selecting pdf which is to be converted to text

In [1]:
```
import tkinter
from tkinter import *
from tkinter.filedialog import askopenfilename
from tkinter.messagebox import showerror

class MyFrame(Frame):
    def __init__(self):
        Frame.__init__(self)
        self.master.title("PDF ANALYSIS FOR CURSE WORDS")
        self.master.rowconfigure(5, weight=1)
        self.master.columnconfigure(5, weight=1)
        self.grid(sticky=W+E+N+S)

        self.button = Button(self, text="Choose  your pdf ", command=self.load_file, width=100)
        self.button.grid(row=1, column=0, sticky=W)

    def load_file(self):
        fname = tkinter.filedialog.askopenfilename(filetypes=(("Pdf files", "*.pdf")
                                        # ,("HTML files", "*.html;*.htm"),
                                         ,("All files", "*.*") ))#used for multiple files selection


        if fname:
            try:
                print("Success")
            except:                     # <- naked except is a bad idea
                showerror("Open Source File", "Failed to read file\n'%s'" % fname)
            return fname

a=MyFrame().load_file()
print(a)
```

```
Success
/home/arjun/13_april/pdf_modules/combined_result.pdf
```

_STEP 2 - converting selected pdf file to text file and saving it in location of current notebook with name "pdf_to*text.txt"*

In [2]:
```
import PyPDF2
pdfFileObj = open(a,'rb')     #'rb' for read binary mode
pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
total_pages=pdfReader.numPages
f= open("pdf_to_text.txt","w+")
print(range(total_pages))
#pdfs=[]
for i in range(total_pages):
    pageObj = pdfReader.getPage(i)          #'9' is the page number
    data=pageObj.extractText()
    f.write(data)
    print(data)
    #print(pdfs[i])
f.close()
```

```
x.2Ftgure I: Two class problemtxs.+-*c+ -.{- o**l' .+ xs1r0754151 Contd...http://www.UPTUonline.comuptuonl
ine.comuptuonline.comuptuonline.com
Wtrat is Parzon window'? Explain. Derive the conditionsfor (r) com'ergence of mean (ii) convergbnce of var
iance'What is X2 test ? Write the significance of hypothesistesting in pattern recognition. Write the uses
of X2 Test'The following table gives the nurnber of accidentthat occurs during the various days of the wee
k'Finrd rvhether the accidents are uniforrnly distributedover the week.Gven: the values of chi square sign
ificance at 5,6,7degrees of freedom are respectively 11'07, 12'59,M.A7 * 5% level of significance.5 Attemp

t anY two of the following :(2x10=20)a) Differentiatebetweenclusteringandclassification. Explaincriteria f
irnction for clustering.rr0754lI Contd...c)http://www.UPTUonline.comuptuonline.comuptuonline.comuptuonlin
e.com
b)Write and orplainK-means cfustering algorithm IllustrateK-means algorithm with the herp ofthe threedimen
sionatdata set of 10 points.grven below :(1, l, 1), (1, l, 2), (1, 3, 2), (2, l, l), (6, 3, l),(6, 4, l),
(6, 6, 6), (6, 6, 7), (6, 7, 6), (7, 7, 7)Consider the initial seeds to be (1, 1, l), (6, 3, l),(6, 6, 6).
what is clustering? Explain. what are different clusteringtechniques? Why is clustering important? What is
anagglomerative clustering algorithm? Explain.15300c)ttws4lhttp://www.UPTUonline.comuptuonline.comuptuonli
ne.comuptuonline.com
```

In [*]:
```
from IPython.core.display import HTML
HTML("<script>Jupyter.notebook.kernel.restart()</script>")
```

Out[3]:

In [ ]:

```

```