Summer 2022 CS 59000-04I AI Software Engr & Application DIS

Welcome Students from **Group 1**

The dataset to be used at the tasks should be mounted from Google Drive for each student.

The datasets are provided as a shared Google Drive folder, if you have not received the link, please inform us.

The instructions for mount the Google Drive folder at the Google Colab and to access the data is:

https://colab.research.google.com/notebooks/io.ipynb

The dataset available are:

- Job postings
- Resumes

If the share "datasets" folder is not showing in Google Colab, follow these instructions to add a shortcut to your drive:

https://support.google.com/drive/answer/2375057

For any follow-up questions/queries:

- Venkata Inukollu inukollv@pfw.edu
- Leandro de Almeida almel01@pfw.edu

Disclaimer: The provided datasets are restrict to be used only during the academic tasks

References:

- Pandas Caculate Statistics/ Summary/ Columns

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

Material/References:

- Pandas - Filter row and columns
- Pandas - Drop multiple columns
- Pandas - Check Pandas data type
- Data - Columns Views - Original Data
- Pandas - Convert value in columns
- Time Ranges/ Time Comparision
- Remove columns or Rows in Pandas
- Remove rows with certain citeria in Python Pandas
- AI BOOKS
- https://towardsdatascience.com/gentle-start-to-natural-language-processing-using-python-6e46c07addf3
- https://monkeylearn.com/keyword-extraction/
- https://www.justintodata.com/use-nlp-in-python-practical-step-by-step-example/
- https://mathdatasimplified.com/

```
# Data Pre-Processing - Job listing Dataset
# Import necessary packages
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import sklearn
import json
import os
import gc # For garbage collection when deal with memory
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

```
os.getcwd()
```

```
'/content'
```

Double-click (or enter) to edit

Job Data Analyst

- Goal: Top 3 Career choices, success factor (example Salaries growth, location, etc as per AI Attributes)

Read Data:

```
# FileNames is a list with the names of the csv files contained in the 'dataset' path
def get_file_names(path):
  filenames = []
  for file in os.listdir(path):
    if file.endswith('.csv'):
      filenames.append(file)
  return filenames

# function that reads the file from the FileNames list and makes it become a dataFrame
def GetFile(fnombre, path):
  location = path + fnombre
  df = pd.read_csv(location)
  return df

file_path_job = './drive/MyDrive/datasets/jobposting/'
# combine all the data frame as one using list complehesion
dfjob = pd.concat([GetFile(file, file_path_job) for file in get_file_names(file_path_job)])
```

```
dfjob.shape
```

```
(300000, 22)
```

Attributes Validation:

```
dfjob['salary_formatted'].value_counts()
```

```
$15 an hour                  2136
From $15 an hour             1441
$15 - $20 an hour            1273
$17 an hour                  1246
$16 an hour                  1188
                             ...
$64,000 - $80,000 a year        1
$15.00 - $17.64 an hour         1
$40 - $75 a day                 1
$3,092 - $3,762 a month         1
$32,176 - $47,316 a year        1
Name: salary_formatted, Length: 20577, dtype: int64
```

```
dfjob['region'].value_counts()
```

```
# Remove because there are all missing value here

     EU          3510
     AS          2761
     SA          1863
     AF           900
     OC           505
     Americas      54
     Name: region, dtype: int64


dfjob['qualifications'].value_counts()
# Can't remove because need this for further basic qualification
# Convert the NAN to 'No requirement'

     ["US work authorization (Required)"]
     2930
     ["US work authorization (Preferred)"]
     2089
     ["High school or equivalent (Preferred)"]
     2001
     ["Driver's License (Required)"]
     973
     ["Bachelor's (Preferred)"]
     960

     ...
     ["HVAC Certification (Required)","US work authorization (Required)","Secret
     (Required)","Associate (Preferred)"]                                       1
     ["Restaurant experience: 4 years (Required)","Day Shift (Preferred)","Night Shift
     (Preferred)"]                                                        1
     ["IT support: 5 years (Required)"]
     1
     ["Microsoft Excel: 4 years (Preferred)","Tableau: 1 year (Preferred)","Data analytics:
     4 years (Preferred)","US work authorization (Preferred)"]       1
     ["UI: 7 years (Preferred)","React: 7 years (Preferred)","Angular: 7 years
     (Preferred)","JavaScript: 7 years (Preferred)"]                           1
     Name: qualifications, Length: 20165, dtype: int64


dfjob['benefits'].value_counts()

     ["Health insurance"]
     5325
     ["Flexible schedule"]
     2717
     ["401(k)","Dental insurance","Health insurance","Paid time off","Vision insurance"]
     1873
     ["Paid time off"]
     908
     ["Dental insurance","Health insurance","Paid time off","Vision insurance"]
     855

     ...
     ["403(b)","403(b) matching","Dental insurance","Flexible spending account","Health
     insurance","Paid time off","Parental leave","Vision insurance"]
     1
     ["401(k)","AD&D insurance","Dental insurance","Disability insurance","Employee
     assistance program","Employee discount","Flexible schedule","Health insurance","Life
     insurance","Paid time off","Referral program","Tuition reimbursement","Vision
     insurance","Wellness program"]      1
     ["401(k)","401(k) matching","Flexible schedule","Health insurance","Health savings
     account","Life insurance","Paid time off","Professional development assistance","Safety
     equipment provided","Tuition reimbursement"]
     1
     ["On-the-job training","Pet insurance","Tools provided","Tuition reimbursement"]
     1
     ["401(k)","AD&D insurance","Commuter assistance","Dental insurance","Disability
     insurance","Employee assistance program","Flexible spending account","Health
     insurance","Health savings account","Paid time off","Parental leave","Tuition
     reimbursement","Vision insurance"]         1
     Name: benefits, Length: 19372, dtype: int64
```

```
# Remove src name
dfjob['srcname'].isnull().sum()
```

```
     193418
```

```
dfjob['country'].value_counts()
```

```
     US    300000
     Name: country, dtype: int64
```

```
dfjob['country_code'].value_counts()
```

```
     US    282417
     BR       800
     GB       681
     CO       668
     CA       612
             ...
     CF         1
     BY         1
     ZM         1
     MD         1
     UG         1
     Name: country_code, Length: 166, dtype: int64
```

```
dfjob['company_name'].value_counts()
```

```
     Deloitte                                3804
     ASSURANCE Independent Agents            1774
     Amazon.com Services LLC                 1401
     Aya Healthcare                          1224
     Soliant                                 1075
                                             ...
     Ardent Counseling Center                   1
     Mobile Management llc                      1
```

```
    Duro Electric                                 1
    CareerStaff Unlimited - Nashville, TN         1
    Sanel Corp                                    1
    Name: company_name, Length: 97715, dtype: int64
```

```
dfjob['company_link'].value_counts()
```

```
    https://www.indeed.com/cmp/The-Est%C3%A9e-Lauder-Companies-1?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54cpcg728qo000&fromjk=00009f127a9e34a7
    1
    https://www.indeed.com/cmp/Holistic-Healing-Collective?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t7pg6pkej800&fromjk=96522f26f3a8fcba
    1
    https://www.indeed.com/cmp/United-Premier?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t62jeq072800&fromjk=96524389f8fbf9ac
    1
    https://www.indeed.com/cmp/Temp-Experts?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t5uj7t48o800&fromjk=965241afe500d938
    1
    https://www.indeed.com/cmp/Kum-&-Go?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t6r8lq051800&fromjk=96523fc6ef1ed652
    1

    ..
    https://www.indeed.com/cmp/Trugreen?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t4ctsj3vu800&fromjk=94750524588fee11
    1
    https://www.indeed.com/cmp/Red-Knight-Solutions,-LLC?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t48h6isah800&fromjk=9475094e35b7bc1f
    1
    https://www.indeed.com/cmp/Bon-Secours?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t48k5i7kk800&fromjk=947510b2566d0887
    1
    https://www.indeed.com/cmp/Beaumont-Health?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54t4aq5h5i1800&fromjk=947512447005a768
    1
    https://www.indeed.com/cmp/Disney-Media-and-Entertainment-Distribution?
    campaignid=vjcmpinfo&from=vjcmpinfo&tk=1g54fb8l5lekk800&fromjk=0710b77c13b3dd2e    1
    Name: company_link, Length: 286348, dtype: int64
```

```
dfjob.head()
```

	jobid	apply_link	company_link
0	00009f127a9e34a7	https://www.indeed.com/applystart?jk=00009f127...	https://www.indeed.com/cmp/The-Est%C3%A9e-Laud...
1	0001783849fce183	NaN	https://www.indeed.com/cmp/H-A-Mapes,-Inc?camp...
2	00027f45e5373e13	https://www.indeed.com/applystart?jk=00027f45e...	https://www.indeed.com/cmp/Accenture?campaigni...
3	00028cda307fcffa	NaN	https://www.indeed.com/cmp/Techo--

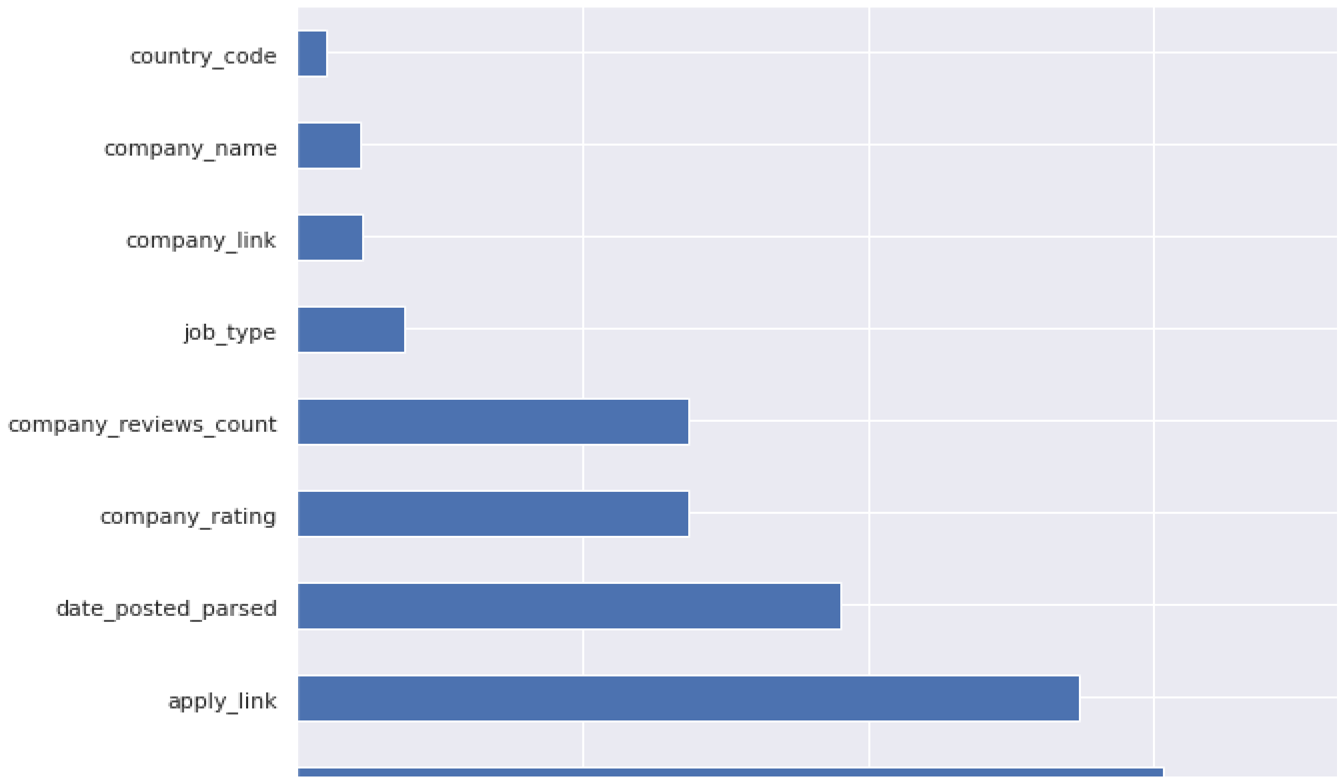
Check Missing Values and Clean Up

Truc Report:

- Data cleaning took me a total of more than 8hrs to looks for the approriate data that need to keep or drop.
- All the attributes need to make sense and support the machine learning model
- Data that consider biased will be drop
- Data that is missing need to fix and transform to meaningful data

```
# Set figure size
plt.rcParams["figure.figsize"]=13,11
sns.set(style='darkgrid')

missing_percentage = dfjob.isna().sum().sort_values(ascending=False)/len(dfjob)
missing_percentage[missing_percentage!=0].plot(kind='barh')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f19c1b3bf90>
```



```
list(dfjob.columns)
```

```
['jobid',
 'apply_link',
 'company_link',
 'company_name',
 'company_rating',
 'company_reviews_count',
 'country',
 'country_code',
 'current_url',
 'date_posted',
 'date_posted_parsed',
 'description',
 'description_text',
 'domain',
 'job_title',
 'job_type',
 'location',
 'region',
 'salary_formatted',
 'benefits',
 'qualifications',
 'srcname']
```

```
dfjob.drop(['jobid','apply_link','company_link','country','current_url','date_posted','date_p

# Out put will be company name and job title
# Remove apply_link because it will not be necessary to have it (we want to analyze the suces
# Apply_link can be removed when the job is filled which is a good sign to analyze these job
# Drop the row where the company name or link is blank:
dfjob.dropna(axis=0, how='all',subset=['company_name', 'job_type'], thresh=2, inplace=True)

# Change null in qualification to no requirement
dfjob['qualifications'] = dfjob['qualifications'].fillna('["No requirement"]')

# Change null in benefits to no benefits
dfjob['benefits'] = dfjob['benefits'].fillna('["No benefits"]')

# Assume all the mssing value in salary_formated is negotiable (50% of the dataset)
dfjob['salary_formatted'] = dfjob['salary_formatted'].fillna('Negotiable')

# Assume all the missing country is Others
dfjob['country_code']=dfjob['country_code'].fillna('Other')

# Fill in the rating with 0
dfjob['company_rating']=dfjob['company_rating'].fillna(0.0)
dfjob['company_reviews_count']=dfjob['company_reviews_count'].fillna(0.0)
```

```
dfjob.shape

# After clean up and drop, we have a new data set of 265633 row and 12attributes
```

```
(265633, 12)
```

Extended Analyst on the company rating and company review

```
# Create norating subset that hold the company doesn't has rating
norating = dfjob.loc[dfjob['company_rating']==0]
```

```
# Validate the result
norating.head()
```

	company_name	company_rating	company_reviews_count	country_code	description	de
1	Harry's Convenience Stores	0.0	0.0	LB	<p>At Harry’s the Store Associate / Foodservic...	At
6	The Michigan Theater Foundation	0.0	0.0	US	<p>The historic Michigan Theater, located in t...	M
11	Facing History and Ourselves, Inc	0.0	0.0	US	<div>\n <p> <b>Position: Director of Developmen...	Po
					<p> <b>Education	

```
# Use value counts to check the name of the company
norating['company_name'].value_counts

# Turn out all the company in this section only post their job one time
```

```
<bound method IndexOpsMixin.value_counts of 1                    Harry's Convenience
Stores
6                The Michigan Theater Foundation
11             Facing History and Ourselves, Inc
16                          The Eye Care Institute
22            Copper Whiskey Bar & Grill, Bozeman
                              ...
29981                                      SunStop
29982    Ford - Lincoln Veteran Careers Program
29988                     Acro Metal Stamping Co.
29992                            Messick and Gray
29997                                  Sanel Corp
Name: company_name, Length: 66460, dtype: object>
```

```
# Validate if there is any rating associate with the number of user
sum(norating['company_rating']==norating['company_reviews_count'])
```

```
66460
```

```
# Check the shape of the dataset
norating.shape
```

```
(66460, 12)
```

Result:

- Due to all the above analyst, I can conclude the norating consist of the company who only post their job one time. So the change to promote in these company is small due to the amount of jobs posted and rating. There for I will remove the row associate with these company where the change is small and the review is none.

```
## Delete norating since we do not need it
del norating
```

```
# Drop the row where the company name or link is blank:
dfjob = dfjob[(dfjob.company_rating != 0) & (dfjob.company_reviews_count != 0)]
```

Double-click (or enter) to edit

```
dfjob.head()
```

	company_name	company_rating	company_reviews_count	country_code	description	des
0	The Estée Lauder Companies	4.0	2214.0	US	<div>\n <p>The Treasury Analyst will assist th...	A
2	Accenture	4.0	21827.0	US	<div> </div>\n<div>\n <div>\n <div>\n <b>ACC...	Fl
3	Techo-Bloc	3.1	114.0	AO	<div>\n Company Description\n <p><b>
 Why W...	D
7	McMahon Associates	3.8	8.0	US	<div>\n <p> <b>POSITION SUMMARY: </b></p> \n <p>...	SU
8	Amazon Kuiper Manufacturing	3.5	82832.0	US	<div>\n <ul>\n <li>BS degree or higher in Ele...	hi

```
dfjob.shape
```

```
     (199173, 12)
```

Double-click (or enter) to edit

```
# Open text file resume
file1 = open('./drive/MyDrive/resume.txt', 'r')
resume_data = []

while True:
    # Get next line from file
    line = file1.readline()
    resume_data.append(line)

    # if line is empty or end of file is reached
    if not line:
        break

file1.close()


# Clean up the resume
import re

# Clean up address, school, name, number, take only character in to the new string list
for i in range(0,len(resume_data)):
    resume_data[i] = re.sub(r'\[.*?\]', '', resume_data[i])
    word1 = " ".join(re.findall("[a-zA-Z]+", resume_data[i]))
    resume_data[i] = word1

# Using the keywords dictionary to hold all the keyword
keyword_dict = []

for line in resume_data:
    li = list(line.split(" "))
    for string_ in li:
        keyword_dict.append(string_.lower()) # Convert the string to lower

# Character that does not necessary to the search can be removed
remove_characters = ['','a','truc','huynh','through','self','classroom','ide','concepts','fou
                     'an','to','on','and','that','this','the','by','in','with','s','of','non'
                     'may','guided','submit','vietnam','cis','any','unsatisfied','services','
                     'customer','ensure','supply','work','year','plans','customer','developin
                     'ensures','supply','options','learn','master','recommendation','science'
                     'previous','concerns','structures','budget','next','methods','stakeholde
                     'visual','higher','coming','teaching','letters','chain','content','tradi
                     'advice','highly','shows','toward','commander','compare','fiscal','direc
                     'ethical','teach','trade']
```

```
soft_skill_remove = ["structure", "experience", "requirements", "worked", "years", "others",
                     "company", "information", "plan", "knowledge", "benefit", "process", "tr
                     "provided", "business", "operation", "systems", "oriented", "level", "ba
                     "reports", "office", "people", "certificate", "pay", "industries", "acco
                     "maintaining", "design", "record", "clients", "bachelor", "projects", "i
                     "meet", "implementation", "sales", "background", "detail", "preparing",
                     "marketing", "result", "weeks", "testing", "financial", "security", "pro
                     "driving", "first", "futures", "instruction", "contracts", "strategies",

for char in remove_characters:
    while(char in keyword_dict) :
        keyword_dict.remove(char)

for char in soft_skill_remove:
    while(char in keyword_dict) :
        keyword_dict.remove(char)

# remove the repeated word in the dictionary
keyword_dict = list(dict.fromkeys(keyword_dict))

# Figure out the length of the keyword dictionaries
len(keyword_dict)
```

```
102
```

```
def pretty_print(word_list):
  index = 1
  for word in word_list:
    print(word, end=', ')
    if index % 10 == 0:
      print('')
    index += 1


# Display list of dictionary
pretty_print(keyword_dict)
```

```
stack, developer, management, analyzing, data, scientist, machine, cyber, introduce, pyt
react, git, docker, web, development, ms, visio, jira, github, slack,
html, css, json, bootstrap, r, shiny, server, framework, flask, restful,
api, javascript, heroku, paas, lifecycle, agile, methodologies, visualization, dashboard
mining, ml, ai, database, sql, mysql, algorithms, c, java, spring,
mvc, net, eclipse, studio, vs, code, anaconda, pycharm, jupiter, notebook,
servlet, apache, tomcat, automation, bot, script, hacking, ui, ux, explaining,
multitask, virtual, machines, windows, linux, mac, osx, autocad, d, modelling,
inventory, planning, forecasting, optimization, logistics, teams, prototype, predict, vo
infrastructure, social, media, compared, logistic, vso, army, medals, vessel, lab,
coding, research,
```

Other dictionary for references

```
# # got these keywords by looking at some examples and using existing knowledge.
# tool_keywords1 = ['python', 'pytorch', 'sql', 'mxnet', 'mlflow', 'einstein', 'theano', 'pys
#  'cassandra', 'aws', 'powerpoint', 'spark', 'pig', 'sas', 'java', 'nosql', 'docker', 'sales
#  'c', 'c++', 'net', 'tableau', 'pandas', 'scikitlearn', 'sklearn', 'matlab', 'scala', 'kera
#  'caffe', 'scipy', 'numpy', 'matplotlib', 'vba', 'spss', 'linux', 'azure', 'cloud', 'gcp',
#  'redshift', 'snowflake', 'kafka', 'javascript', 'qlik', 'jupyter', 'perl', 'bigquery', 'un
#  'scikit', 'powerbi', 's3', 'ec2', 'lambda', 'ssrs', 'kubernetes', 'hana', 'spacy', 'tf', '
#  'seaborn', 'mllib', 'github', 'git', 'elasticsearch', 'splunk', 'airflow', 'looker', 'rapi
#  'jquery', 'nodejs', 'd3', 'plotly', 'bokeh', 'xgboost', 'rstudio', 'shiny', 'dash', 'h20',
#  'hive', 'cognos', 'angular', 'nltk', 'flask', 'node', 'firebase', 'bigtable', 'rust', 'php
#  'kubeflow', 'rpython', 'unixlinux', 'postgressql', 'postgresql', 'postgres', 'hbase', 'das
# # added r packages doesn't seem to impact the result
#  'dplyr','ggplot2','esquisse','bioconductor','shiny','lubridate','knitr','mlr','quanteda','
#  'leaflet','janitor','ggvis','plotly','rcharts','rbokeh','broom','stringr','magrittr','slid
#  'rmysql','rsqlite','prophet','glmnet','text2vec','snowballc','quantmod','rstan','swirl','d

# # another set of keywords that are longer than one word.
# tool_keywords2 = set(['amazon web services', 'google cloud', 'sql server'])

# # hard skills/knowledge required.
# skill_keywords1 = set(['statistics', 'cleansing', 'chatbot', 'cleaning', 'blockchain', 'cau
#  'dashboard', 'geospatial', 'ocr', 'econometrics', 'pca', 'gis', 'svm', 'svd', 'tuning', 'h
#  'salesforcecom', 'segmentation', 'biostatistics', 'unsupervised', 'supervised', 'explorato
#  'recommender', 'recommendations', 'research', 'sequencing', 'probability', 'reinforcement'
#  'chi', 'knn', 'outlier', 'etl', 'normalization', 'classification', 'optimizing', 'predicti
#  'clustering', 'cluster', 'optimization', 'visualization', 'nlp', 'c#',
#  'regression', 'logistic', 'nn', 'cnn', 'glm',
#  'rnn', 'lstm', 'gbm', 'boosting', 'recurrent', 'convolutional', 'bayesian',
#  'bayes'])

# # another set of keywords that are longer than one word.
# skill_keywords2 = set(['random forest', 'natural language processing', 'machine learning',
#  'time series', 'nearest neighbors', 'neural network', 'support vector machine', 'computer
#  'text analytics', 'power bi', 'a/b testing', 'ab testing', 'chat bot', 'data mining'])

# degree_dict = {'bs': 1, 'bachelor': 1, 'undergraduate': 1,
#                'master': 2, 'graduate': 2, 'mba': 2.5,
#                'phd': 3, 'ph.d': 3, 'ba': 1, 'ma': 2,
#                'postdoctoral': 4, 'postdoc': 4, 'doctorate': 3}


# degree_dict2 = {'advanced degree': 2, 'ms or': 2, 'ms degree': 2, '4 year degree': 1, 'bs/'
#                 '4-year degree': 1, 'b.s.': 1, 'm.s.': 2, 'm.s': 2, 'b.s': 1, 'phd/': 3, 'p
#                 'm.s/': 2, 'm.s./': 2, 'msc/': 2, 'master/': 2, 'master\'s/': 2, 'bachelor\
# degree_keywords2 = set(degree_dict2.keys())
```

Sampling

The amount of words is too large, I will reduce the data to 10,000 row so that we can see how our model works on the smaller scale (sample is 5% of the data)

```
sample_size=10000

# Create a sample of 10,000 rows
dfjob_s = dfjob.sample(n=sample_size)

# Validate the transactions
dfjob_s.head()
```

	company_name	company_rating	company_reviews_count	country_code	descri
21931	Applebee's	3.5	17274.0	US	<div>\n <p>< Turn every wo into
15884	Papa John's Pizza | SARPJ	3.5	13133.0	US	< <b>SUMMARY< <br> Che pr
2450	Sunriver Resort	4.0	68.0	US	<div>\n < <div>\n < Sun
5510	Delaware County Intermediate Unit	3.3	41.0	US	<p>The DCIU a reading spe </p
14552	Glenwood Regional Medical Center	2.7	716.0	US	<p></p>\n< < <p>POS SUMMA

```
# Has to import NLTK and download averaged_perceptron_tagger
import nltk
from nltk import pos_tag
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize

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

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
tags = pos_tag(keyword_dict)
```

```
tags_list = []
for tag in tags:
  tags_list.append(tag[1])

tags_list = list(dict.fromkeys(tags_list))

# Figure out the length of the keyword dictionaries
len(tags_list)
```

```
8
```

```
tags_list
```

```
['NN', 'VBG', 'NNS', 'JJ', 'IN', 'VBP', 'RB', 'VBN']
```

```
dfjob_s.shape
```

```
(10000, 12)
```

Apply POS tagging

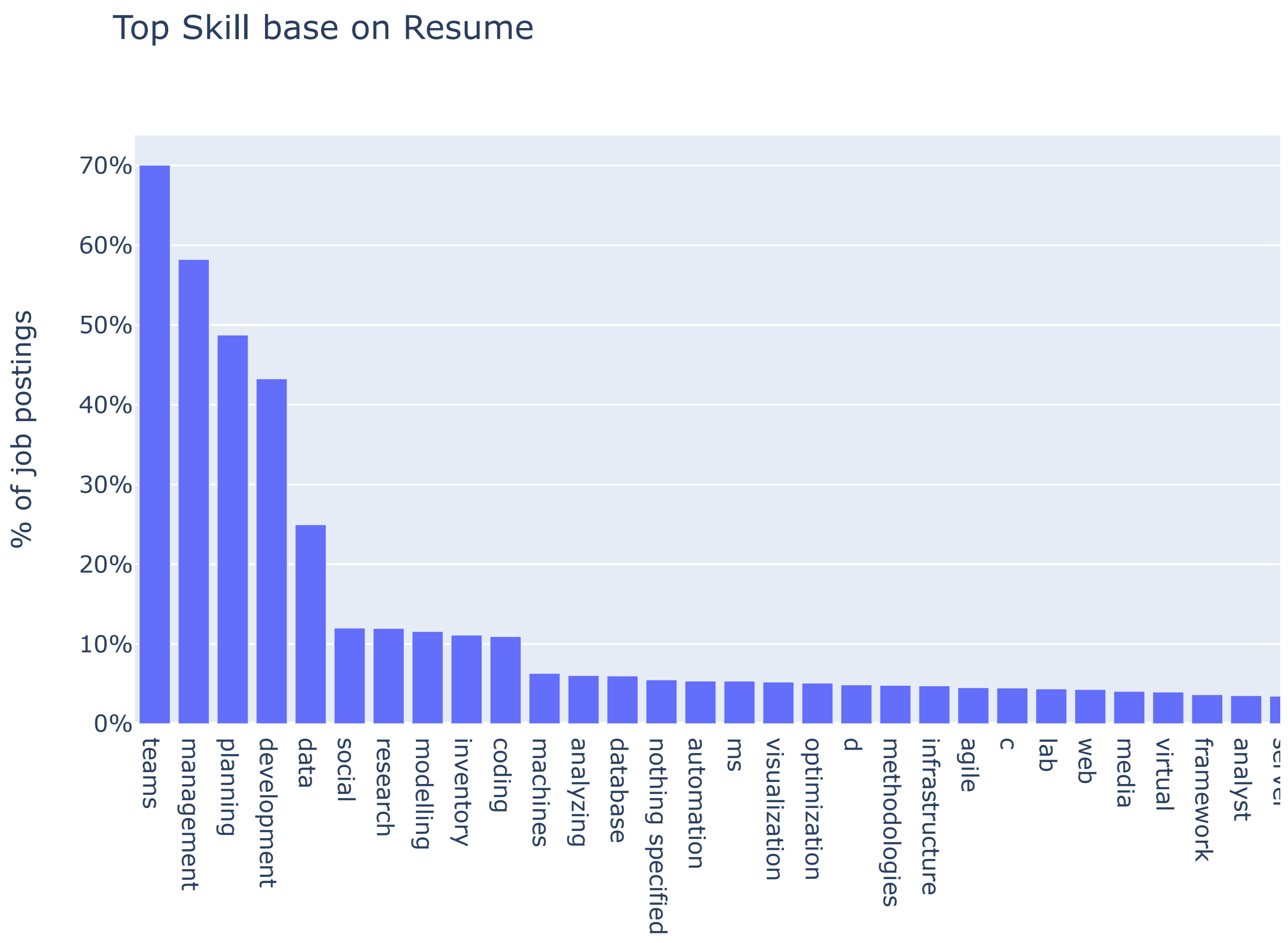
```
ps = PorterStemmer()
```

```
# process the job description.
def prepare_job_desc(desc):
    # tokenize description.
    tokens = word_tokenize(desc)

    # Parts of speech (POS) tag tokens.
    token_tag = pos_tag(tokens)

    # Only include some of the POS tags.
    include_tags = ['VBN', 'VBD', 'JJ', 'JJS', 'JJR', 'CD', 'NN', 'NNS', 'NNP', 'NNPS']
    filtered_tokens = [tok for tok, tag in token_tag if tag in include_tags]

    # stem words.
    stemmed_tokens = [ps.stem(tok).lower() for tok in filtered_tokens]
    return set(stemmed_tokens)


dfjob_s['job_description_word_set'] = dfjob_s['description_text'].map(prepare_job_desc)

# process the keywords
tool_keywords_set = set([ps.stem(tok) for tok in keyword_dict]) # stem the keywords (since th
tool_keywords_dict = {ps.stem(tok):tok for tok in keyword_dict} # use this dictionary to reve


pretty_print(tool_keywords_set)
```

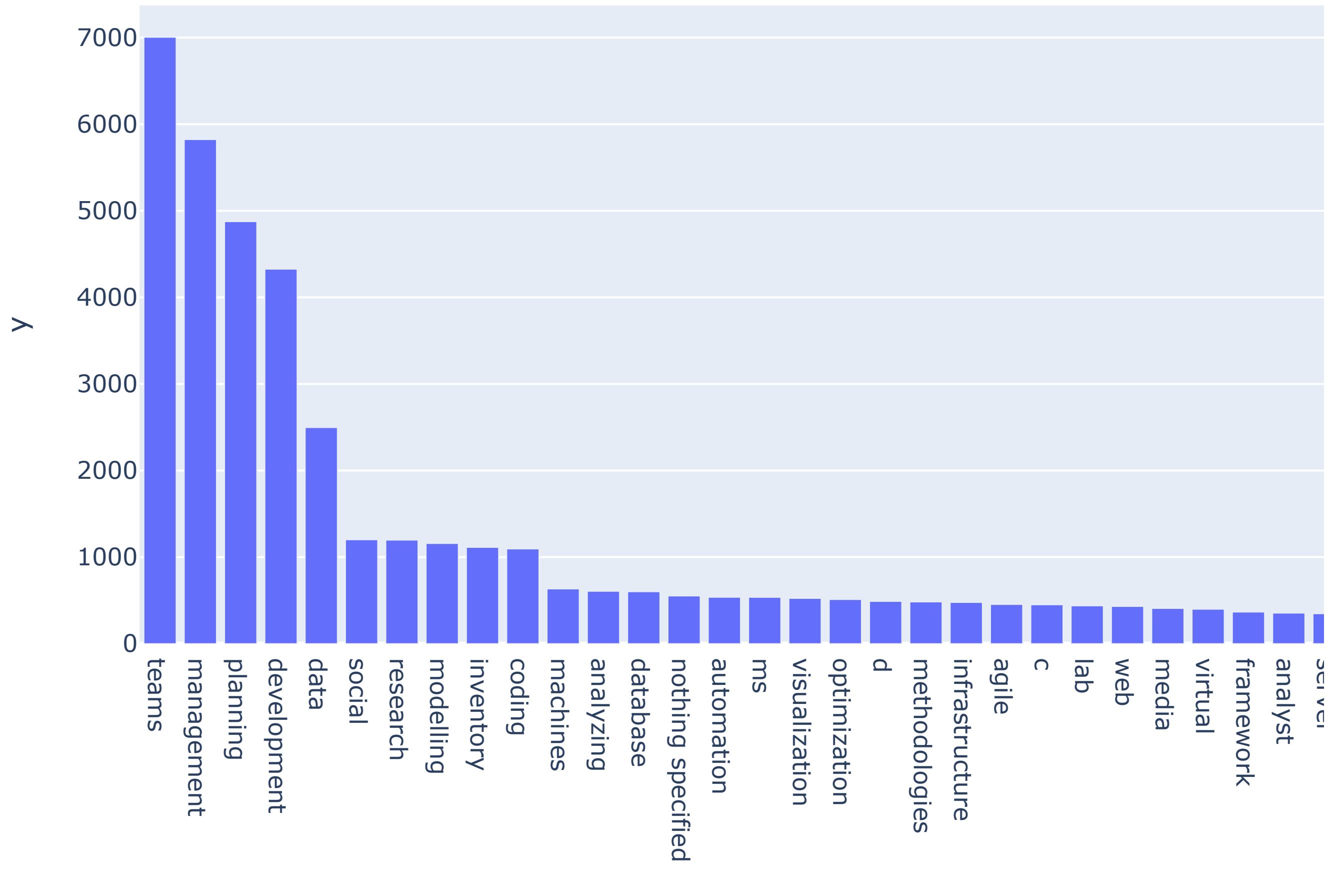
```
     net, lifecycl, react, flask, visual, json, virtual, server, paa, plan,
     autom, ms, window, code, github, c, bootstrap, armi, mine, introduc,
     osx, mysql, python, git, studio, lab, model, team, html, logist,
     spring, ux, analyz, d, predict, notebook, ai, agil, data, css,
     script, mvc, mac, develop, ui, infrastructur, multitask, explain, volatil, compar,
     stack, scientist, algorithm, sql, ml, analyst, api, social, javascript, machin,
     rest, inventori, heroku, forecast, linux, vso, medal, web, prototyp, docker,
     vs, vessel, framework, hack, slack, bi, apach, databas, pycharm, tomcat,
     eclips, shini, visio, manag, anaconda, jupit, methodolog, r, servlet, jira,
     cyber, dashboard, java, bot, autocad, optim, media, research,
```

```
tool_list = []

msk = dfjob_s['country_code'] != '' # just in case you want to filter the data.

num_postings = len(dfjob_s[msk].index)
for i in range(num_postings):
    job_desc = dfjob_s[msk].iloc[i]['description_text'].lower()
    job_desc_set = dfjob_s[msk].iloc[i]['job_description_word_set']

    # check if the keywords are in the job description. Look for exact match by token.
    tool_words = tool_keywords_set.intersection(job_desc_set)

    # label the job descriptions without any tool keywords.
    if len(tool_words) == 0:
        tool_list.append('nothing specified')

    tool_list += list(tool_words)


# create the list of tools.
df_tool = pd.DataFrame(data={'cnt': tool_list})
df_tool = df_tool.replace(tool_keywords_dict)

df_tool_top = df_tool['cnt'].value_counts().reset_index().rename(columns={'index': 'tool'}).i
```

```
from plotly import __version__
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
import plotly.graph_objs as go

# visualize the tools.
layout = dict(
    title='Top Skill base on Resume',
    yaxis=dict(
        title='% of job postings',
        tickformat=',.0%',
    )
)

fig = go.Figure(layout=layout)
fig.add_trace(go.Bar(
    x=df_tool_top['tool'],
    y=df_tool_top['cnt']/num_postings
))

iplot(fig)
```



```
import plotly.express as px

fig = px.bar( x=df_tool_top['tool'],
    y=df_tool_top['cnt'])
fig.show()
```



Find most align

Using the list just found and find the most

```
pretty_print(df_tool_top['tool'])
```

```
    teams, management, planning, development, data, social, research, modelling, inventory,
    machines, analyzing, database, nothing specified, automation, ms, visualization, optimiz
    infrastructure, agile, c, lab, web, media, virtual, framework, analyst, server,
    windows, sql, logistic, script, lifecycle, forecasting, python, java, r, restful,
    stack, api, predict, spring, javascript, scientist, compared, studio, linux, dashboard,
    bi, jira, prototype, algorithms, ai, react, net, cyber, explaining, visio,
df_tool_top['grow_percentage'] = df_tool_top['cnt']/sample_size
```

```
keyword_list_after_clean = list(df_tool_top['tool'])
```

```
pretty_print(keyword_list_after_clean)
```

```
teams, management, planning, development, data, social, research, modelling, inventory,
machines, analyzing, database, nothing specified, automation, ms, visualization, optimiz
infrastructure, agile, c, lab, web, media, virtual, framework, analyst, server,
windows, sql, logistic, script, lifecycle, forecasting, python, java, r, restful,
stack, api, predict, spring, javascript, scientist, compared, studio, linux, dashboard,
bi, jira, prototype, algorithms, ai, react, net, cyber, explaining, visio,
```

Result:

- According to the key word list and the percentage, we can extract many information such as:
  - Softskill: teams, management, planning, development
  - Hardskill: data, modelling, research, inventory, social, coding, analyzing, machines, optimization, database, visualization, methodologies, automation, virtual, infrastructure, media, lab, web, framework, script, agile, forecasting, server, logistic, lifecycle
  - Pogramming languages: Python, r, spring, javascript, scientist, compared, studio, linux, dashboard, bi, jira, prototype, algorithms, ai, react, net, cyber, explaining, visio,

Support Functions

```
# Garage collections
gc.collect()
```

```
1167
```

✓ 0s completed at 3:09 PM