Open in Colab

Resume transformation

Original Resume

```
Truc Huynh
[jackyhuynh87@gmail.com]
[https://www.linkedin.com/in/trucdev/]
[https://github.com/jackyhuynh]
Experienced full-stack developer, project management & coordination through different industries with a strong technical background and analyzing skills.
EDUCATION
Master of Science, Computer Science [Purdue University Fort Wayne, Fort Wayne, IN]
Certificate in Data Scientist & Machine Learning [North Carolina State University]
Bachelor of Science, Computer Science [ECPI University, Newport News, VA]
Associate in Cyber Security [De Anza College]
Introduce to Self-Driving Car Certificate [Udacity.com]
Python, React, Git, Docker, web development: self-learn [Udemy, HackerRank, Codility, and DataCamp]
SKILLS
Proficient with Office 365, MS Project, MS Visio, Jira, Git, GitHub, and Slack
Experience in web application development with (HTML, CSS, JSON, Bootstrap, R Shiny, Python web server framework, Flask, RESTful API, and JavaScript, Heroku (PaaS))
Experience in the software development lifecycle, Agile methodologies, data visualization, dashboard design (BI)
Classroom experience in data mining, machine learning, ML/AI (Python, R), database technologies (SQL, MySQL), data analytics, data structures & algorithms (Python, C/ C++, Java), framework
Experience in relationship building and explaining technical concepts to non-technical audiences. Highly collaborative, team-oriented, and pay attention to detail with the ability to multit
Working knowledge with Virtual machines, Windows, Linux, Mac OSX, AutoCAD, 3D Modelling
Experience with inventory planning and supply chain management (forecasting, planning, optimization & logistics)
EXPERIENCE
Co-founder & Program Manager: [7 Figures Trader]
Build an online training program to teach people to trade (stock, options, futures contracts). Work and strategize with others on cross-functional teams and stakeholders to design the web a
Co-founder & Business Analyst: [Luxe Nails & Spa L.L.C]
Lead cross-functional decision-making & implementation to define infrastructure plan, business support systems, and operation plan. Provided advice and instruction on work methods, practice
Logistic Specialist & Watercraft Engineer: [US Army • Fort Eustis, VA]
Developing an inventory record and tracking database that ensures VSO warehouse equipment and inventory are 100% accountable. This database was evaluated to benefit my company in years (Arm
Teaching Assistant: [De Anza College• Cupertino, CA]
Worked directly under the CIS Lab Manager toward any coding issues that students may submit. Guided students' projects to ensure they meet their instructor's requirements
Information Technology Support Specialist: (Vietnam)
Provided base-level IT support to clients. Identified, developed, installed, and monitored clients’ systems to ensure their efficiency. Conducted research to address customers’ concerns tha
```

Transformation Resume:

```
[79] (dfjob_s_2['bag_of_words'].iloc[1000])
```

```
'stack developer management analyzing data scientist machine cyber introduce python react git docker web development ms visio jira github slack html css json bootstrap r shiny s
erver framework flask restful api javascript heroku paas lifecycle agile methodologies visualization dashboard bi mining ml ai database sql mysql algorithms c java spring mvc ne
t 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 volatility analyst infrastructure social media compared logistic vso army medal
s vessel lab coding research'
```

Report Model I: POS Tokenization and Keywords Filtering

Summary:

POS Tokenization and Keywords Filtering aims to filter keywords in the resume and match them with tokenized job_description (after clean up, tidy, token stored in data lake). The matching result is stored in another data frame for further advising and career recommendations.

Data Tidy:

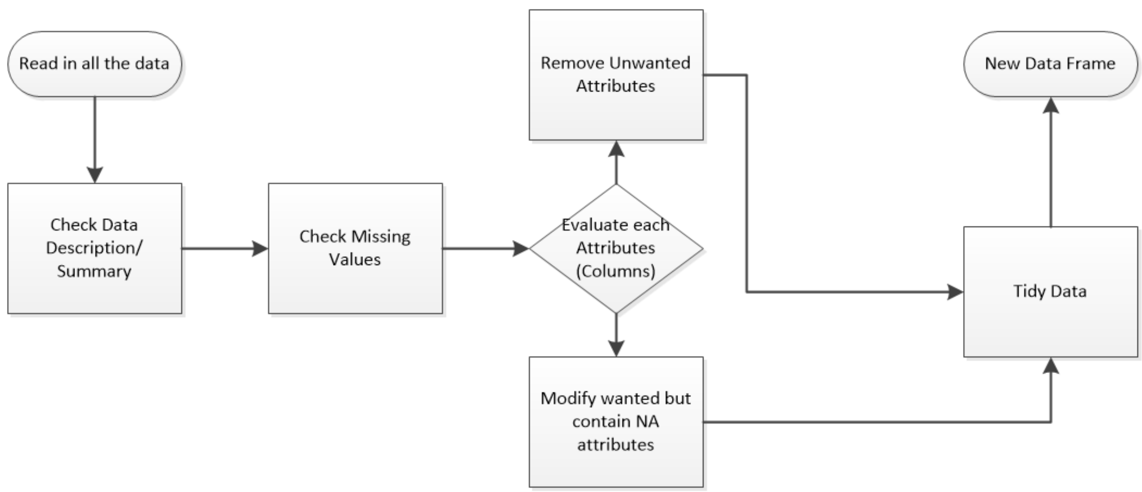
Process:

Tidy Data process in the following steps:

- Data Description: quick check of data type, statistic, etc.
- Check missing values
- Evaluate attributes
- Remove unwanted attributes:

- Drop columns: 'jobid', 'apply_link', 'company_link', 'country', 'current_url', 'date_posted', 'date_posted_parsed', 'domain', 'region', 'srcname' because they will not relevant to our ML Model
- Drop rows: where the company name or job title is blank (no point to keep); where company and reviews count is 0

- Fix NA attributes:
  - Change 'null': in the 'benefits' columns to 'no benefits'
  - Change 'null' in the 'qualifications' columns to 'no qualifications'
  - Change 'null' in the 'salary_formated' to 'neogotiable'
  - Change 'null': in the 'country_code' to 'other'
- Tidy Data
  - Remove more than 100,000 rows and 10 columns.



Data after Tidy Summary:

Name this data tidy version I, we will also use this for our 2nd Model

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
company_name	199173	48175	Deloitte	3746	NaN	NaN	NaN	NaN	NaN	NaN	NaN
company_rating	199173.0	NaN	NaN	NaN	3.564244	0.502832	1.0	3.3	3.6	3.9	5.0
company_reviews_count	199173.0	NaN	NaN	NaN	3945.074357	12875.904782	2.0	44.0	288.0	1906.0	236369.0
country_code	199173	155	US	187555	NaN	NaN	NaN	NaN	NaN	NaN	NaN
description	199173	168669	<div>\n Assurance believes that you're unique,...	874	NaN	NaN	NaN	NaN	NaN	NaN	NaN
description_text	199173	168450	Assurance believes that you're unique, and you...	875	NaN	NaN	NaN	NaN	NaN	NaN	NaN
job_title	199173	123554	Assistant Manager	438	NaN	NaN	NaN	NaN	NaN	NaN	NaN
job_type	199173	186	["Full-time"]	132319	NaN	NaN	NaN	NaN	NaN	NaN	NaN
location	199173	28117	Remote	2850	NaN	NaN	NaN	NaN	NaN	NaN	NaN
salary_formatted	199173	14595	Negotiable	132510	NaN	NaN	NaN	NaN	NaN	NaN	NaN
benefits	199173	13073	["No benefits"]	142203	NaN	NaN	NaN	NaN	NaN	NaN	NaN
qualifications	199173	9070	["No requirement"]	173994	NaN	NaN	NaN	NaN	NaN	NaN	NaN

NLP Processing:

After Data Tidy, and Resume Tidy (Transformation), then NLP will be processed in the following steps:

Streamlining the Job Descriptions using NLP Techniques:

- Step 1: Part of Speech (POS): tagging keywords list (which filter from candidate resume) with indetified tags from NLTK package

```
[('stack', 'NN'),
 ('developer', 'NN'),
 ('management', 'NN'),
 ('analyzing', 'VBG'),
 ('data', 'NNS'),
 ('scientist', 'NN'),
 ('machine', 'NN'),
 ('cyber', 'NN'),
 ('introduce', 'NN'),
 ('python', 'NN'),
 ('react', 'NN'),
 ('git', 'JJ'),
 ('docker', 'NN'),
 ('web', 'NN'),
 ('development', 'NN'),
 ('ms', 'NN')
```

- Step 2: Using the found tags as filter to filter out all the unrelated tags which means the words without these below tags will be removed from job_description
  ```
  # In this case the list of tags is
  include_tags = ['VBN', 'VBD', 'JJ', 'JJS', 'JJR', 'CD', 'NN', 'NNS', 'NNP', 'NNPS']
  ```
- Step 3: Tokenizing the Job Descriptions: parsing the text string into different sections by applying the filter in step 2 (include_tags)
- Step 4: Steaming the words: The stemming process allows computer programs to identify the words of the same stem despite their different look (e.g. “models”, and “modeling” both have the same stem of “model”)

```
{'agil': 'agile',
 'ai': 'ai',
 'algorithm': 'algorithms',
 'anaconda': 'anaconda',
 'analyst': 'analyst',
 'analyz': 'analyzing',
 'apach': 'apache',
 'api': 'api',
 'armi': 'army',
 'autocad': 'autocad',
 'autom': 'automation',
 'bi': 'bi',
 'bootstrap': 'bootstrap',
 'bot': 'bot',
 'c': 'c',
 'code': 'coding',
```

- Step 5:Lowercasing the words Sample of job description after transformation

```
# Sample print ot of job description at index 12 after transformation

pretty_print(dfjob_s_1['job_description_word_set'].iloc[12])
```

```
experi, happi, 5:00pm, nation, written, patern, travel, equival, record, verbal,
prefer, next, transport, color, real, servic, maintain, test, except, strong,
statement, blood, skill, manner, environ, opportun, center, flexibl, vital, eoe/aa,
live, certif, data, reliabl, log, race, patient, cleric, entri, accredit,
superior, prepar, employ, abl, career, age, payment, profession, great, day,
duties/respons, 8:30a-, duti, requir, time, minim, healthi, diploma, veteran, religion,
techniqu, equal, necessari, track, work, statu, practic, need, standard, comfort,
organ, differ, inform, specimen, due, monday-friday, school, blind, person, perform,
role, peopl, phlebotomist, abil, offic, sexual, proud, face, drug, addit,
venipunctur, proven, site, commun, schedul, continu, client, chanc, phlebotomi, custom,
screen, job, alcohol, pleas, applic, privaci, team, gender, accur, develop,
high, provid, ident, capillari, clean, drive, challeng, group, bill, agenc,
health, more, previou, process, labcorp, collect, passion, step, growth, sex,
administr, origin, orient, disabl, compani, analysi, today, supervis,
```

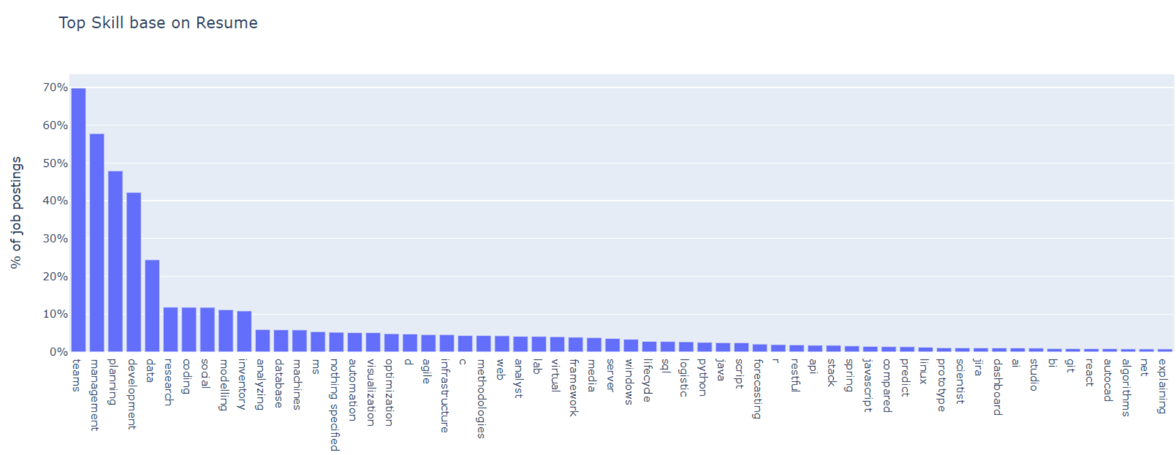
Run Model & Result

- Using Set (Python Data Structures) to return the similarity of each job compare to the skill set in the resume.The frequency list will be create by the amount of time a words (skill) appear in each job and combine them to caluclate the over all percentage.There is none repeated words (cause we use Set data structure)

```
df_tool_top.head()
```

	tool	cnt	grow_percentage
0	teams	7001	0.7001
1	management	5818	0.5818
2	planning	4718	0.4718
3	development	4289	0.4289
4	data	2545	0.2545

Data Visualization:

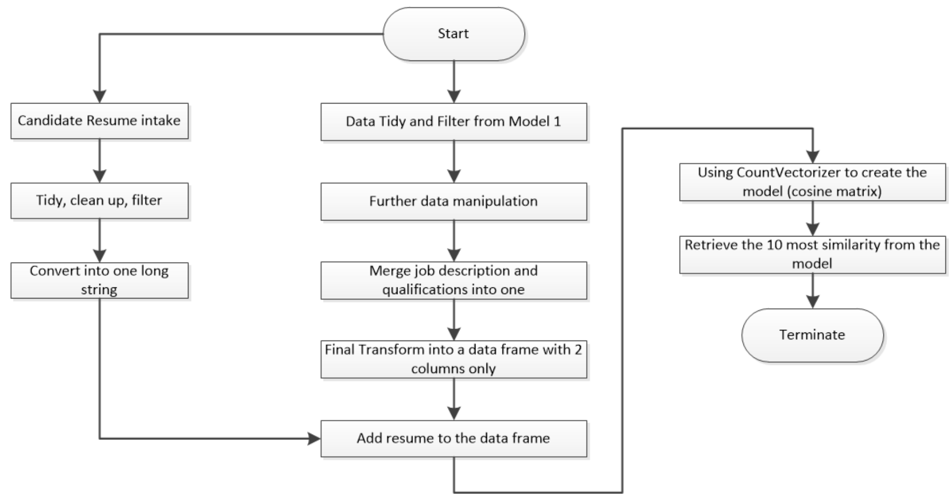


Deploy application:

- Check Demo Indeed_Demo
- App is deployed at Heroku.

Model 2: Bags of Words and Cosine Similarity

BAG OF WORDS APPROACH



Summary:

- Bags of Words and Cosine Similarity merge all the columns into two columns. The idea is to merge all the related information to the job search into one column (or attributes) and convert them to vectors using cosine similarity.
- Measure the distance between resume's vector and all other job description vector using cosine similarity

Data Tidy:

NLP Process:

Tidy Data in Model 2 process in the following steps:

- Use the tidy data that create from Model 1 (before any NLP process for Model 1 and keep working)
- Further Data Transformation:
  - Drop columns: 'country_code', 'description', 'job_type', 'salary_formatted', 'benefits' due to not related to Model 2
- Streamlining the Job Descriptions using NLP Techniques:
  - Tokenizing the Job Descriptions: parsing the text string into different sections
  - Part of Speech (POS): tagging the job description
- Data Transformation:
  - Merge 'company_name', 'company_rating', 'company_reviews_count', 'job_title', 'location' in to one name 'jobs_all_information'
  - Merge the tokenized 'job_description' and 'qualifications' into one name 'bag of word'

Final Data Frame Transformation

	jobs_all_information	bag_of_words
0	Access Staffing LLC|3.8|251.0|Financial Manage...	[relationship respons analyz strong complianc ...
1	State of Utah|3.6|242.0|Human Resources Analys...	[hr 35.98 employe offic parti supplement exper...
2	Intel|4.1|5638.0|LTD Manufacturing Integration...	[comput tomorrow deep transform jmp event futu...
3	Hertz|3.3|6767.0|Financial Analyst	[visibl employe consist concept relev offic cr...
4	Amazon Data Services, Inc.|3.5|82832.0|Cluster...	[visibl matur offic deep multi-ten infrastruct...
...	...	...
9995	NorthStar Anesthesia|2.3|31.0|Mon Health Medic...	[ob/cv gynecolog facil rang psychiatri joint c...
9996	Werum|3.7|147.0|Software Engineer	[document comput factori concept question join...
9997	Big Sky Managed Care|3.8|6.0|Certified Pharmac...	[employe montana 60 good 17.00 fridayexperi fi...
9998	WakeMed|3.9|671.0|Patient Care Tech, OR (full-...	[employe heart workforc facil identity/express...
9999	General Dynamics|3.8|2387.0|Administrative Ass...	[employe enterpris inteladminjob relev tomorro...

10000 rows × 2 columns

Create Similarity Matrix:

[58]

```
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer

# Transform the
count = CountVectorizer()
count_matrix = count.fit_transform(dfjob_s_2['bag_of_words'])

# Create the matrix to compare
cosine_sim = cosine_similarity(count_matrix, count_matrix)

# create a Series of job titles, so that the series index can match the row and column index of the similarity matrix.
indices = pd.Series(dfjob_s_2['jobs_all_information'])

# Validate similarity matrix
print(cosine_sim)
```

```
[[1.         0.26436075 0.22018561 ... 0.21331048 0.220802   0.        ]
 [0.26436075 1.         0.25171085 ... 0.22678252 0.23238252 0.        ]
 [0.22018561 0.25171085 1.         ... 0.17360548 0.27422143 0.00577379]
 ...
 [0.21331048 0.22678252 0.17360548 ... 1.         0.24587594 0.00690263]
 [0.220802   0.23238252 0.27422143 ... 0.24587594 1.         0.00799565]
 [0.         0.         0.00577379 ... 0.00690263 0.00799565 1.        ]]
```

Result:

```
recommend('new_candidates_resume', cosine_sim)
```

```
['Crescent Bank|3.3|165.0|Web Developer',
 'CACI|3.8|2086.0|Sr. Software Engineer',
 'Open Systems Technologies Corporation|4.3|4.0|Full Stack Java Developer',
 'Deloitte|4.0|10699.0|AI Center of Excellence - Platform AVP, Software Engineer',
 'Leidos|3.7|1326.0|.NET Software Applications Developer',
 'Deloitte|4.0|10699.0|Full Stack Java Developer',
 'WELLS FARGO BANK|3.7|42353.0|Senior Systems Operations Engineer',
 'BRS|3.6|47.0|Programmer/Analyst III',
 'Qcentrio|3.0|3.0|Full Stack software developer',
 'Seneca Resources|4.1|19.0|Java Developer']
```