Personal Career Manager Software Agent

User Guide

Team Name: Team10

Alfred Tay Wenjie, Kenneth Goh Chia Wei, Tan Heng Han, Wang Zilong, Raymond Ng Boon Cheong

NUS-ISS Master of Technology in Intelligent Systems

ISY5001 Machine Reasoning and Reasoning Systems

Contents

Executive Summary

The Level-Up Application connects employees, employers and training providers. It helps employees discover their career aspirations and maps out a career path for them. It matches employees to open job postings and recommends relevant training courses to users of the application. The backend system enabling Level-Up's functionality is an Intelligent Reasoning System made up of multiple integrated modules performing Knowledge Discovery, Business Rules Reasoning and Career Path Optimisation. User testing was carried out with 13 test users and the feedbacks received has validated the business value of the Level-Up Application.

Project Scope

Opportunity Statement

i. **Unclear Career Aspiration**. In conversations with friends and colleagues, it was a common observation that most people do not think about their career aspiration and those who has career aspirations often does not know how to achieve their career aspirations in the shortest time.

ii. **Rapid Changes in Job's Core Skills**. According to a Linkedin Report[1], by 2020, it is expected that 42 percent of the core skills required for a job will change. This implies that many professionals will have to acquire relevant new skills in order to stay competitive in the job market. With the myriad of course offerings under the SkillsFuture initiative and other training providers, working professionals face a sea of choices and does not know exactly which training course to give priority.

iii. **Sub-optimal Employee to Job Allocation**. At the Global or even just Country or City level, the current employee to job allocation is not optimised. The argument for this is the observation that in many organisations, an employee's promotion not just depend on his readiness for the next higher position but also depends on whether there is a vacant higher position to be filled.

Proposed Solution

The proposed solution is a platform that brings together the employees, the employers and the training providers. It churns out a personalised career roadmap based on a person's employment history and career aspiration. For people who are unsure about their career aspirations, the system will ask guiding questions to help them discover their career aspirations. The system then assesses the user's readiness for the next higher position and searches its job database for suitable job opportunities. Based on the user's personalised career roadmap and current job market demand (data from job posting details), the system recommends highly relevant courses from its course database. From the employee user group's point of view, the solution can be viewed as a sort of personal career manager. From employer and training provider's point of view, the system helps them to find suitable employees and trainees.

[1] Future of Skills 2019: Anticipating what's next for your business, Linkedin Talent Solutions

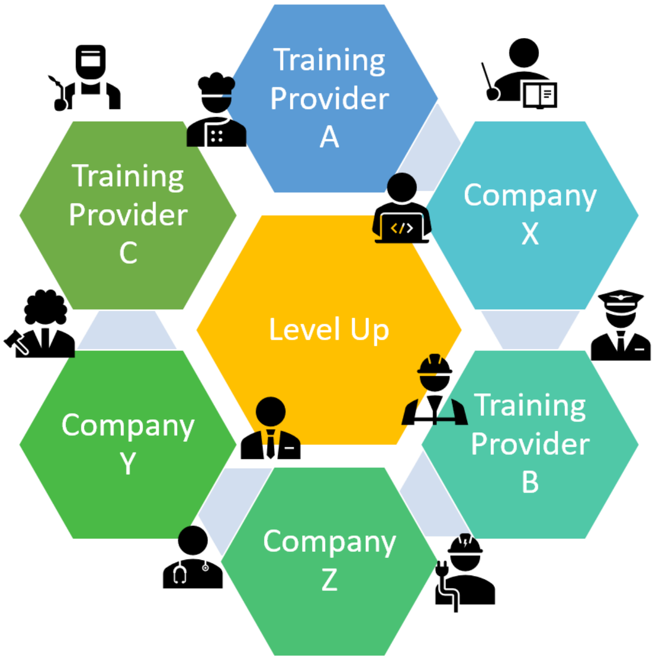


Figure 2b

Value Proposition

The value proposition is using Machine Reasoning techniques to build an Intelligent Reasoning System which helps the users discover their career aspirations and supports them in achieving their career aspirations in the shortest possible time. It also works towards better optimisation of employee to job allocation. Consider an example below:

Jonathan is ready for the next higher position but there is no such needs or vacancy in his current organisation. However, there is a vacancy in another organisation which is having trouble finding a suitable person to fill the similar higher position. Jonathan left his company to take up this opportunity. Shortly after, a more junior person filled up the vacancy left behind by Jonathan. In this example, everybody wins.

Solution Overview

The solution comprised multiple integrated modules performing Knowledge Discovery, Business Rules Reasoning and Career Path Optimisation. Starting from Knowledge Discovery, Apriori Algorithm is used to filter and extracted possible career movements from employment history data. Bag of word model and clustering technique is used to extract job competencies and group together similar job titles based on job description. Business Rule Reasoning and Fuzzy Logic is used to guide users in discovering their career aspirations and to recommend relevant training courses. Career Path Optimisation is accomplished using ASTAR Search and Genetic Algorithm. The input to the system is a user questionnaire and the output is a Career Roadmap, a list of Job Recommendations and Training Course Recommendations (if applicable). Figure 3 shows the overall data flow of the system.

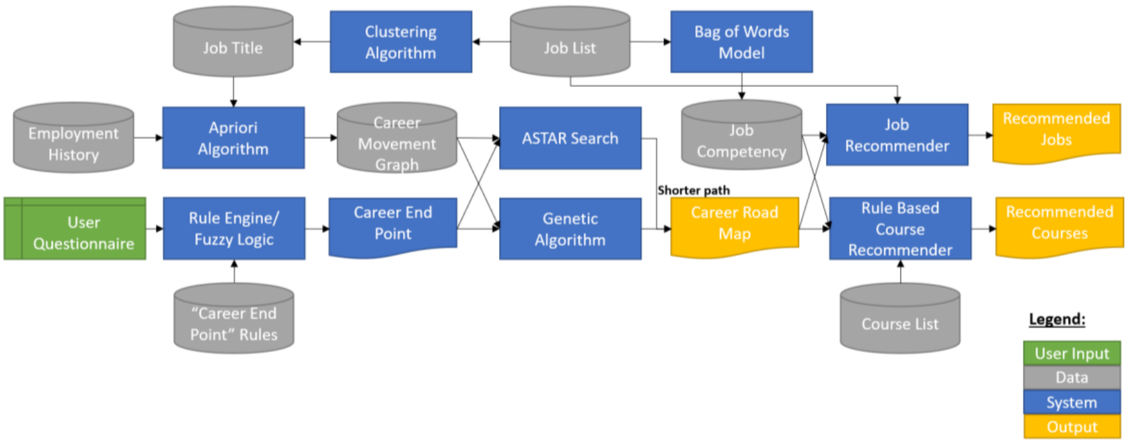


Figure 3

Knowledge Discovery and Data Mining for Career Movement

The purpose of knowledge discovery in individual Curriculum Vitae (CV) was to extract its entire career path data that included the time spent to move from job position A to B and its intermediate job position titles. The career path data of all sampled CVs served as part of the data mining process to identify patterns that augmented our understanding and aided us in deciding which unusual data was to be removed.

The initial scope was to include seventeen job titles where each job title served as a node in the knowledge graph. However, the scope was expanded subsequently, which would be explained later, to include forty-five job titles. Two tables that formed the knowledge base were produced to capture node to node and heuristic costs.

Knowledge Discovery and Data Mining Process

Figure 4a shows 3 stages in the Knowledge Discovery and Data Mining process: Data pre-processing, Knowledge Discovery and Data post-processing for Career Movement.

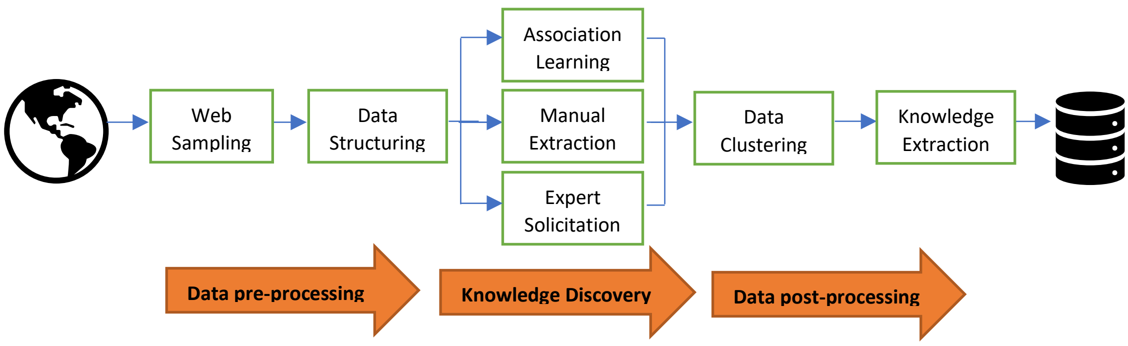
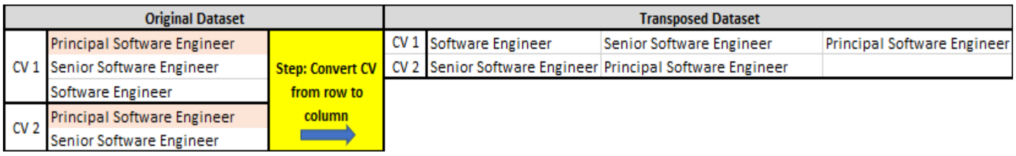


Figure 4a

Data Pre-processing

CVs and career path information were sourced from public websites[2] to fit into the knowledge base. Features of sampled data were manually extracted to ensure data was structured and only columns relevant to calculating node to node and heuristic costs were specified. The columns included the job position and its accumulated years of experiences. Figure 4b showed that this dataset was also used to create the transposed dataset for knowledge discovery. A python script to transpose dataset was implemented and its technical details were found in Annex A.



Original Dataset	
CV 1	Principal Software Engineer
	Senior Software Engineer
	Software Engineer
CV 2	Principal Software Engineer
	Senior Software Engineer

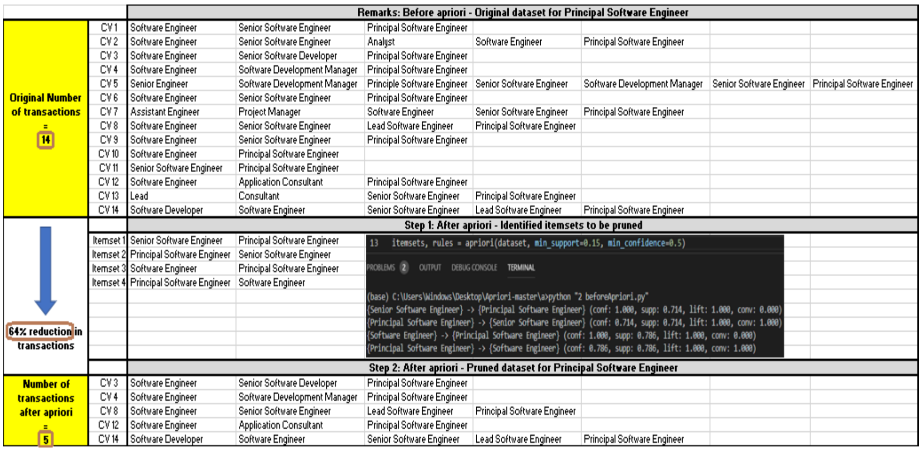
Transposed Dataset			
CV 1	Software Engineer	Senior Software Engineer	Principal Software Engineer
CV 2	Senior Software Engineer	Principal Software Engineer	

Figure 4b

Knowledge Discovery

Three techniques were employed in knowledge discovery stage. It involved association learning, manual extraction and expert solicitation.

1. Association Learning. Apriori algorithm was applied on the transposed dataset to identify association between nodes using support, confidence and lift scores. Each job title dataset contained approximately ten to fourteen transactions. A support value of 0.15, confidence level of 0.5 and lift rating of 1 were imposed to identify item sets that have weak association. These item sets found in the transactions were pruned. Figure 4c-1 showed a 64% reduction in associative data after Apriori. A python script using Apriori algorithm was implemented and its technical details were found in Annex A.



Remarks: Before apriori - Original dataset for Principal Software Engineer

Original Number of transactions = 14

CV 1	Software Engineer	Senior Software Engineer	Principal Software Engineer				
CV 2	Software Engineer	Senior Software Engineer	Analyst	Software Engineer	Principal Software Engineer		
CV 3	Software Engineer	Senior Software Developer	Principal Software Engineer				
CV 4	Software Engineer	Software Development Manager	Principal Software Engineer				
CV 5	Senior Engineer	Software Development Manager	Principle Software Engineer	Senior Software Engineer	Software Development Manager	Senior Software Engineer	Principal Software Engineer
CV 6	Software Engineer	Senior Software Engineer	Principal Software Engineer				
CV 7	Assistant Engineer	Project Manager	Software Engineer	Senior Software Engineer	Principal Software Engineer		
CV 8	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer			
CV 9	Software Engineer	Senior Software Engineer	Principal Software Engineer				
CV 10	Software Engineer	Principal Software Engineer					
CV 11	Senior Software Engineer	Principal Software Engineer					
CV 12	Software Engineer	Application Consultant	Principal Software Engineer				
CV 13	Lead	Consultant	Senior Software Engineer	Principal Software Engineer			
CV 14	Software Developer	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer		

Step 1: After apriori - Identified itemsets to be pruned

64% reduction in transactions

Itemset 1	Senior Software Engineer	Principal Software Engineer
Itemset 2	Principal Software Engineer	Senior Software Engineer
Itemset 3	Software Engineer	Principal Software Engineer
Itemset 4	Principal Software Engineer	Software Engineer

```
13    itemsets, rules = apriori(dataset, min_support=0.15, min_confidence=0.5)

PROBLEMS 2   OUTPUT   DEBUG CONSOLE   TERMINAL

(base) C:\Users\Windows\Desktop\Apriori-master\a>python "2 beforeApriori.py"
{Senior Software Engineer} -> {Principal Software Engineer} (conf: 1.000, supp: 0.714, lift: 1.000, conv: 0.000)
{Principal Software Engineer} -> {Senior Software Engineer} (conf: 0.714, supp: 0.714, lift: 1.000, conv: 1.000)
{Software Engineer} -> {Principal Software Engineer} (conf: 1.000, supp: 0.786, lift: 1.000, conv: 0.000)
{Principal Software Engineer} -> {Software Engineer} (conf: 0.786, supp: 0.786, lift: 1.000, conv: 1.000)
```

Step 2: After apriori - Pruned dataset for Principal Software Engineer

Number of transactions after apriori = 5

CV 3	Software Engineer	Senior Software Developer	Principal Software Engineer		
CV 4	Software Engineer	Software Development Manager	Principal Software Engineer		
CV 8	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer	
CV 12	Software Engineer	Application Consultant	Principal Software Engineer		
CV 14	Software Developer	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer

Figure 4c-1

[2] CV Sampling - https://linkedin.com, Career Path - https://www.zippia.com, https://hrtrendinstitute.com, https://www.codefellows.org/blog/what-success-developer-looks-like, https://www.nextiva.com/blog/cio-career-path.html#s5

2. Manual Extraction. Human expertise was applied on the entire CV transactions to eliminate abnormalities that might cause biases during the computation of heuristics and node to node costs. An example of abnormalities observed was a person who started working as an engineer and subsequently named himself as Director after establishing his own company. In such scenario, the transaction was removed. Figure 4c-2 illustrated the results of a pruned dataset after human intervention.

	Remarks: Before manual extraction - Original dataset for Principal Software Engineer							
Before pruning	CV 1	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 2	Software Engineer	Senior Software Engineer	Analyst	Software Engineer	Principal Software Engineer		
	CV 3	Software Engineer	Senior Software Developer	Principal Software Engineer				
	CV 4	Software Engineer	Software Development Manager	Principal Software Engineer				
	CV 5	Senior Engineer	Software Development Manager	Principle Software Engineer	Senior Software Engineer	Software Development Manager	Senior Software Engineer	Principal Software Engineer
	CV 6	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 7	Assistant Engineer	Project Manager	Software Engineer	Senior Software Engineer	Principal Software Engineer		
	CV 8	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer			
	CV 9	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 10	Software Engineer	Principal Software Engineer					
	CV 11	Senior Software Engineer	Principal Software Engineer					
	CV 12	Software Engineer	Application Consultant	Principal Software Engineer				
	CV 13	Lead	Consultant	Senior Software Engineer	Principal Software Engineer			
	CV 14	Software Developer	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer		
	Step 1: Manual Extraction - Identified itemsets to be pruned for Principal Software Engineer due to data abnormalities							
Identified itemsets to be removed	CV 2	Software Engineer	Senior Software Engineer	Analyst	Software Engineer	Principal Software Engineer		
	CV 4	Software Engineer	Software Development Manager	Principal Software Engineer				
	CV 5	Senior Engineer	Software Development Manager	Principle Software Engineer	Senior Software Engineer	Software Development Manager	Senior Software Engineer	Principal Software Engineer
	CV 7	Assistant Engineer	Project Manager					
	CV 10	Software Engineer	Principal Software Engineer					
	CV 12	Software Engineer	Application Consultant	Principal Software Engineer				
	CV 13	Lead	Consultant					
	Step 2: After manual extration - Pruned dataset for Principal Software Engineer							
After pruning	CV 1	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 3	Software Engineer	Senior Software Developer	Principal Software Engineer				
	CV 6	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 7	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 8	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer			
	CV 9	Software Engineer	Senior Software Engineer	Principal Software Engineer				
	CV 11	Senior Software Engineer	Principal Software Engineer					
	CV 13	Senior Software Engineer	Principal Software Engineer					
	CV 14	Software Developer	Software Engineer	Senior Software Engineer	Lead Software Engineer	Principal Software Engineer		

Figure 4c-2

a. Expert Solicitation. A combination of web extraction and expert interview was employed to record a typical career path for various pinnacle job titles such as Chief Information Officer, Chief Technology Officer and etc. This information served as a guide to gather general career path that might have otherwise been pruned by machine learning and manual extraction due to possible lack of dataset and human misrepresentation of career path respectively.

b. Data Clustering. Figure 4c-3 showed the results of the three techniques that led to a selection of datasets for data post-processing stage through human assessment.

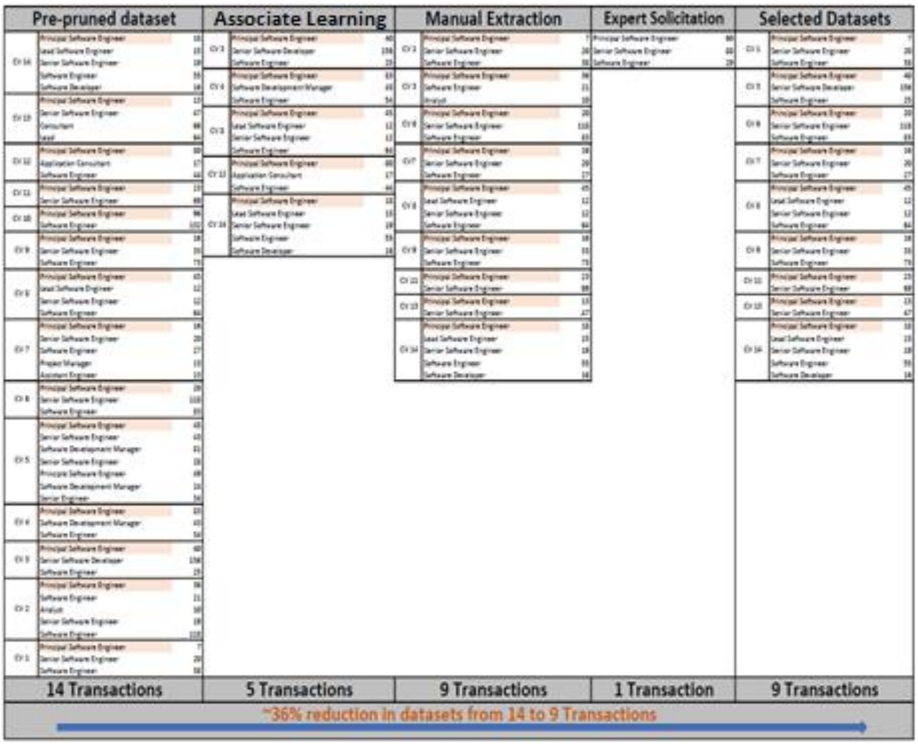


Figure 4c-3

Data Post-processing

While working on the inputs for knowledge graph, it was observed that the total unique job titles available in the entire CV transactions have expanded from seventeen to approximately two hundred. The method to handle this will be discussed in the next section.

Heuristic and node to node costs was computed before ingesting into the knowledge base. A python script to calculate heuristic and node to node cost was implemented and its technical details were found in Annex A.

i. Heuristic cost. It is the estimated time required to reach a particular end job position by taking the average time taken from the first job to reach the say end job position.

ii. Node to node cost. It is the average time taken to move from a job position to another connected job position.

Knowledge Discovery and Data Mining for Job Competency Job Title

The purpose of knowledge discovery in Job Competency and Job Title is to extract Job Competencies hidden in lengthy job descriptions and to group together similar Job Titles.

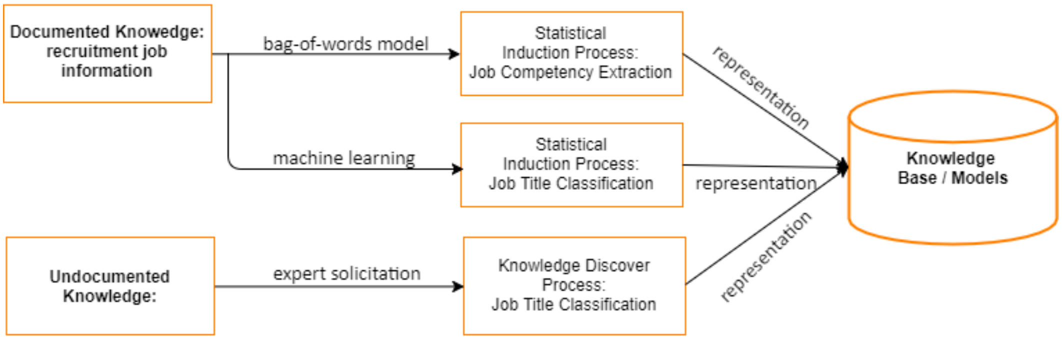


Figure 5-1

SQLiteExpert[3] database management tool is used to import all the acquired knowledge into organised Knowledge Base.

SQLiteExpert database administration

Figure 5-2

Job Competency Extraction - Bag-of-words model

i. Introduction

The bag-of-words[4] model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity.

ii. Why we do?

Job competency is a key criterion for matching job and course recommendation. In most job posting data, job competency is listed among other job requirements and an extraction method is required to automatically extract them.

iii. How we do?

Most of the job description and job requirement for IT industry are well defined, many skills are listed to describe a specific technical domain. How to accurately extract these skills? We found a Technology Dictionary[5] (Figure 5a-1) which has 14,131 IT-related words which we use to build a bag-of-word model to extract Job Competencies from the job descriptions. We

[3] SQLiteExpert http://www.sqliteexpert.com/

[4] Wikipedia Bag-of-words model https://en.wikipedia.org/wiki/Bag-of-words_model#N-gram_model

[5] Techopedia IT dictionary https://www.techopedia.com/dictionary

assume the (frequency of) occurrence of each word is an indicator to show how important a certain competency is.

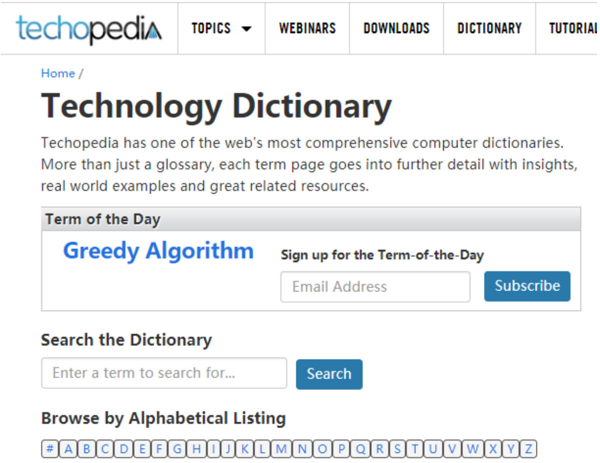


Figure 5a-1

iv. Pre-processing:

- Load job description data previously collected from public website[6].
- Tokenize text data
- Filter away stop words
- Delete punctuation mark (skillsets such as C++, C# and .NET are kept)

v. N-gram model extracting:

We apply the N-gram model because words like natural, language and processing are not specific skills individually, but are combined into a trigram: NATURAL LANGUAGE PROCESSING, it is a technical skill. We need to ensure these skills are included in the bag-of-words by using bi-grams and tri-grams features.

- Generate N-gram word list from job description data
- Generate N-gram competency list from Technology Dictionary
- Search competency from job description and generate bag-of-words
- Save to excel file

vi. Frequency Filter:

As time complexity is too high when running searches, a threshold is set to filter away those less frequent competencies mentioned in job description before running the search. We assume these less frequent competencies, in general, are less important, and frequently mentioned words are important job competency that companies require.

```
for key in newdict_1.keys():
    if threshold < newdict_1.get(key): # filter frequency
        if key in divider('terminology_modified_1.xlsx', 1):
# search competency
            key_1.append(key)
```

[6] JobStreet https://www.jobstreet.com.sg/

vii. Running & Result:

Here are parameters need to set when running competency_extraction.py file:

```python
wb = load_workbook('competency_extraction_data.xlsx') # load excel file
ws = wb["sheet1"] # load excel sheet
threshold = 5 # set filter
save = 'competency_extraction_result.xlsx'# save result
```

Annex C is the result for software engineer job competency.

Job Title Classification: Clustering Algorithm, tf-idf indexing and Manual Grouping

i. Challenge

When performing Job Association Analysis, we focus on how people change from one job position to another. Our data are mainly from public CV & resume and the job titles in these CV & resume are various. For example, staffing level, different people use different tile to describe entry-level, junior, assistant, associate, senior, principal, manager, etc. As for the job category, we have words such as technical, sale, management, business, product, project, etc. The combinations of job titles are huge, we have 185 job titles from the raw data and many of them are just a different name for the same job. We need to group similar Job Titles together in order to build a generalised knowledge base that could be applied to the anticipated variety of input Job Titles.

ii. Solution

We need to combine these job titles into more generic and representative job titles. Base on job title related information (i.e. job description and job requirement), we apply unsupervised classification method to discover the natural clusters in the Job Titles.

iii. Data preparation

One column for job titles, one column for job-related text information.

iv. Tf-idf Indexing.

Tf-idf Indexing is applied on the job-related text information before passing into the KMeans clustering algorithm.

```python
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(max_features=200000,
                                   stop_words='english',
                                   use_idf=True,
                                   max_df=0.8,
                                   min_df=0.2,
                                   ngram_range=(1,2))
tfidf_matrix = tfidf_vectorizer.fit_transform(y)
```

v. k-means Classification

```python
from sklearn.cluster import KMeans
num_clusters = 4    # manually defined according to actual need
km = KMeans(n_clusters=num_clusters)
km.fit(tfidf_matrix)
```

vi. Result & Interpretation

When setting num_cluster = 4 (Figure 5b-1 left), we notice that all the analyst jobs are clustered into one group, software engineer job one group, software management related jobs one group, non-software management jobs one group. Even without data cleaning and hyper-parameter tuning, we can clearly see those red high-lighted Top Keywords in Figure 5b-2 make sense from each group.

When setting num_cluster = 3 (Figure 5b-1 right), we can see the change is clustering all the software related job into one group no matter it is management position or technical position. it still makes sense.

num_cluster = 4

	title	cluster
0	data analyst junior	1
1	data analyst senior	1
2	business analyst	1
3	software engineer junior	3
4	software engineer senior	3
5	software manager	2
6	software director	2
7	project manager	0
8	sales manager	0

num_cluster = 3

	title	cluster
0	data analyst junior	1
1	data analyst senior	1
2	business analyst	1
3	software engineer junior	0
4	software engineer senior	0
5	software manager	0
6	software director	0
7	project manager	2
8	sales manager	2

Figure 5b-1

Cluster	Job Title	Top Keywords
1	data anglyst junior data anglyst senior buiness analyst	Cluster 1: analytics work location data analysis excel personal caed caed com mining view personal data analyst reports business requirements banking privacy policy data analyst location address policy privacy view larger
3	software engineer junior software engineer senior	Cluster 3: work location software engineer programming engineer solution engineers code web location address consulting platform real time application development design develop oracle scalable ai architecture cloud improve
2	software manager software director	Cluster 2: architecture leadership sap enterprise dynamics iot cloud solution vendor review business applications marketing architect objectives application development lead cyber strategic drive pay
0	project manager sales manager	Cluster 0: risk ll ey area solution clients professional better risk management progress pmo certified consulting digital plan long pre sales budget project manager pmp

Figure 5b-2

vii. limitation

The above is a demonstration of how clustering could be used to group job titles together. Due to lack of job description data for all the 185 job titles, we have largely performed manual grouping based on our own knowledge to group the 185 Job Titles into 45 Job Titles[7].

[7] Refer to Annex B – List of Job Titles

Business Rules Reasoning

Determining Career End Point

Basic Rules

Career end goal is one that is difficult to be generalised in a few simple steps. In our project, we use Myers-Briggs Type Indicator [8](MBTI), a personality type assessment that is commonly used by schools and organisations like UC Berkeley, McKinsey e.g. to determine what kind of teams could be or conflicts could arise from different personalities.

Orientation to World	*Take in Information*	*Make Decisions*	*Take in Info. or Decide*
Extraverted Energized by others *or* **Introverted** Energized by ideas, emotions, memories	**Sensing** Using five senses *or* **Intuition** Using gut or instincts	**Thinking** Logical, problem solvers *or* **Feeling** Consider others, compassionate	**Perceiving** Taking in information *or* **Judging** Organizing information and making decisions

Figure 5a

With our project, we try to simulate the potential use of such personality indicators in our application to help the users better understand their own personality and preference of their work behavior, which will then aid in formulating a better career goal for themselves.

According to studies[9], individuals who are comfortable taking charge of and motivating others tend to be Extroverted (E), Intuitive (N), and Thinking (T). Conversely, individuals who prefer to do the job instead of directing others tend to be Introverted (I), Sensing (S), and Feeling (F) individuals.

Using the above as our key indicator of preference of doing management work or preference for becoming a domain knowledge expert, we build a fuzzy logic scoring system to weigh the preference of user for Management Roles using both his MBTI Personality and a direct question of preference. An example of the scoring table is show below:

MANAGEMENT SCORE	
E	3
N	3
T	3
J	0
I	-2
S	-2
F	-2
P	0
PREFERENCE QNS	
YES	**9**
NO	**-6**

[8] https://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/home.htm?bhcp=1

[9] Buboltz, W. C., Johnson, P., Nichols, C., Miller, M. A., & Thomas, A. (2000). MBTI Personality Types and SII Personal Style Scales. Journal of Career Assessment, 8(2), 131–145. https://doi.org/10.1177/106907270000800203

Table 5a-1

An example of the scoring being done is shown below:

ENTJ			
E	3		
N	3		
T	3		
J	0	**WEIGHTS**	**SCORE**
	9	0.4	3.6
YES	9	0.6	5.4
		TOTAL	**9**
		MANAGEMENT	

Table 5a-2

Fuzzy Logic Augmentation

Our decision table is a measuring of the score above the average being suitable for the management role and below the average, the user will be shown the non-top management roles.

MAX. SCORE	**9**
MIN. SCORE	**-4.8**
AVERAGE	**2.1**

Table 5a-3

CONDITION	TOTAL SCORE >= 2.1	TOTAL SCORE < 2.1
TOP MANAGEMENT ROLES	**T**	**F**
NON-TOP MANAGEMENT ROLES	**F**	**T**

Table 5a-4

Based on a second study[10], we map preference with higher weight of 0.6 as locus of control positively correlates to an individual's career goal engagement. Thus, weighing higher on their individual preferences outwardly and innately allows the users to take control of their career goals.

Secondly, in line with the preference of the user taking priority, we allow them to choose between the career types of Balanced, Sales vs Technical in IT Roles. This allows them to better control and select their own career goal to reach.

[10] Haratsis, J. M., Hood, M., & Creed, P. A. (2015). Career Goals in Young Adults: Personal Resources, Goal Appraisals, Attitudes, and Goal Management Strategies. Journal of Career Development, 42(5), 431–445. https://doi.org/10.1177/0894845315572019

Course Recommendation

Course recommendation generally consists of Executive Education courses provide by Institute of Systems Science, NUS[1]. Each course is added as an entry in the Django database and courses are individually tagged with the skills that are covered in the course.

For the rule-based system implementation, each course is assigned to a rule which performs a match of the skills tagged to the course with the input facts. The input facts consist of a set of skills which are determine as lacking from the user required for the user to achieve his or her career goals. Each course rules which are fired will be append to a list and will be displayed in the result page.

Career Path Optimisation

Based on the user's current job position and aspired job position, the Career Path Search module uses Informed Search and Evolutional Computing techniques to search for a shortest path between these 2 job positions.

Career Movement Graph

The possible career paths are represented in a career movement graph with each node representing a job position. Directional links between nodes represent possible career moves from one job position to another job position. Labels on links represent the average number of months it takes to move from 1 job position to another job position. The same career movement graph is used by both ASTAR Search and Genetic Algorithm.

ASTAR Search

A custom ASTAR Search algorithm is developed to search the career movement graph for the shortest career path. The estimated future path cost, h(n), is defined as the difference in the average number of months to reach a certain position, between the current search node and the aspired job position. For example, if the aspired job position is Chief Technology Officer (takes on average 146 months to reach from a fresh grad) and the current search node is Project Manager (takes on average 88 months to reach from a fresh grad), h(n) would be 58 months (146 months – 88 months).

In this design and application of the ASTAR Search algorithm, finding the global optimal is not guaranteed as there may exist a path between 2 job positions which takes lesser than h(n). However, during testing the ASTAR Search algorithm computes quickly and returns reasonably short paths.

Genetic Algorithm

A custom Genetic Algorithm is also developed from scratch to compliment the ASTAR Search by introducing randomness in traversing the search space. Each chromosome is a possible career path starting with the current job position. Unlike typical Genetic Algorithm implementations, chromosomes representing career paths can and will have different length. The fitness function is the total cost of the career path and a penalty cost is imposed if the aspired career position is not in the career path. Roulette wheel selection method is used. A single-point crossover is used as the length of the career path is generally short and crossover can only happen when a legitimate move exists (link exist in career movement graph). To increase the chance of successful crossover, the chromosome is matched with multiple chromosomes until a legitimate move is found. For mutation operation, a node is randomly selected to pursue a different career path.

It was observed during testing that in contrast with the ASTAR Search algorithm, the Genetic Algorithm does not return the same result every time. On a few occasions, it has outperformed the ASTAR search in finding a shorter career path.

The final solution runs both the ASTAR Search and Genetic Algorithm and returns the shorter of the 2 paths found.

System Architecture

General system architecture centers around the Django Server Framework which host 2 rules base system, 2 search algorithm modules and a built-in Sqlite3 database.

Framework for the rule-based expert system is provided by the Expera[2] expert system library. Implementation of both ASTAR Search and Genetic Algorithms are adapted from pseudo code.

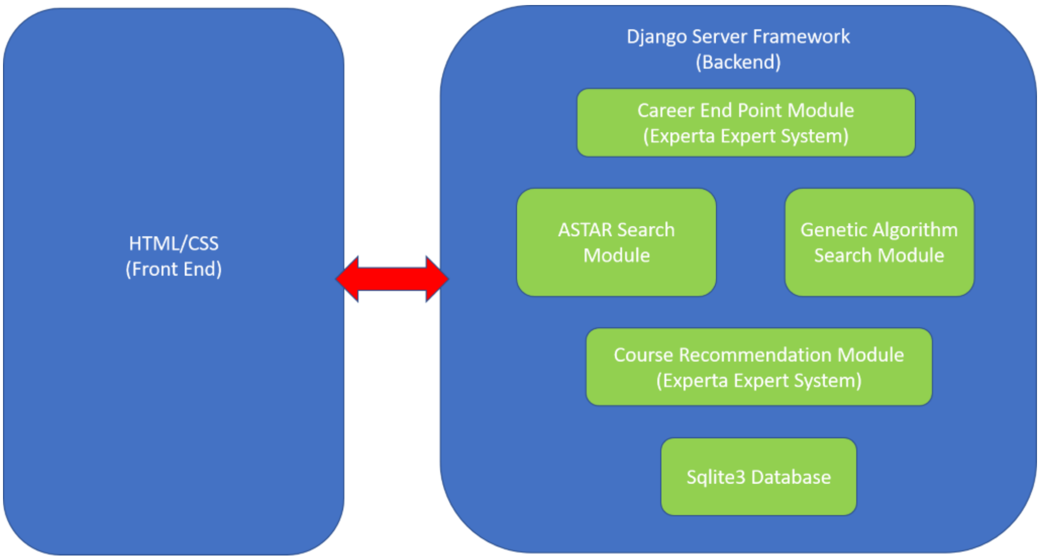


Figure 7

Database Design

The database used is a Sqlite3 database in-built into the Django server. Field and value types indicated in the database diagram below are built-in model types from the Django framework.

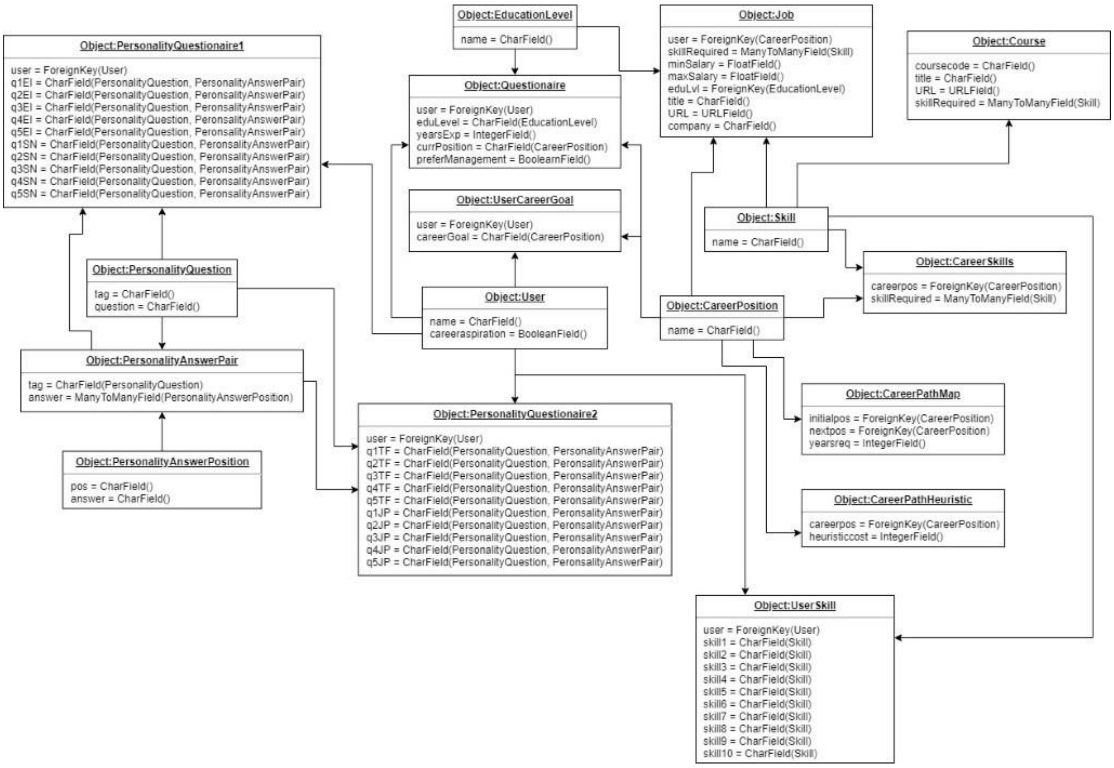


Figure 8

User Testing and Survey

Survey Form

Table 9a showed the set of questionnaires that was designed to capture the usability of our platform and areas for improvement.

Question	Response	Purpose
How likely is it that you would recommend the Career Recommendation Platform to a friend or colleague?	- Very likely - Likely - Maybe - Unlikely - Very unlikely	Marketability of our platform
How useful you find the Career Recommendation Platform?	1 - Lowest, 10 – Highest	Measurement of user friendliness
How easy/intuitive is it to interact with the Career Recommendation Platform?	- Very Easy to use - Easy to use - Neither Easy nor difficult to use - Difficult to use - Very difficult to use	

Which feature is the most important to you?	- Career Path Proposal - Job Availability - Course Recommendation - None of the above	Area for improvement
How can the Career Recommendation Platform be improved to serve you better?	Open ended	

Table 9a

Survey Results

Table 9b showed the results of the thirteen responses. It was interpreted as users were opened to the concept of a career recommendation system and found it relatively intuitive to manoeuvre in our platform. It was also observed that most of the users found the career recommendation feature meaningful. Further, this was supported by the high percentile awarded to the career path proposal feature that revealed our business value proposition met the objective of assisting user to achieve their career aspirations. The responses to the open-ended question suggested area of improvements where features such as adding new IT job roles in the career graph and uploading of CV file could be incorporated to support future demands and expectations in our platform.

Question	Response	Results
How likely is it that you would recommend the Career Recommendation Platform to a friend or colleague?	- Very likely - Likely - Maybe - Unlikely - Very unlikely	36.1% - Very likely 35.8% - Likely 28.1% - Maybe 0% - Unlikely 0% - Very unlikely
How useful you find the Career Recommendation Platform?	1 - Lowest, 10 – Highest	23.1% - Rating 10 30.8% - Rating 9 7.7% - Rating 8 23.1% - Rating 7 15.4% - Rating 6
How easy/intuitive is it to interact with the Career Recommendation Platform?	- Very Easy to use - Easy to use - Neither Easy nor difficult to use - Difficult to use - Very difficult to use	53.8% - Very Easy to use 46.2% - Easy to use 0% - Neither Easy nor difficult to use 0% - Difficult to use 0% - Very difficult to use
Which feature is the most important to you?	- Career Path Proposal - Job Availability - Course Recommendation - None of the above	61.5% - Career Path Proposal 30.8% - Job Availability 7.7% - Course Recommendation 0% - None of the above

How can the Career Recommendation Platform be improved to serve you better?	Open ended	Results were summarized in section 9b

Table 9b

Conclusion

In the successful implementation of the Level-Up solution, the team has creatively applied machine reasoning techniques which works together to enable the end-to-end functionality and user experience. In applying and implementing these techniques, the team has gained a deeper understand of the respective topics.

Below are a listing of the limitations and future enhancements.

Limitations

i. Given limited time and resources, the current dataset only contains traditional Information Technology jobs.

Future Enhancements

i. Expand the dataset to cover jobs in all major industries

ii. Develop a functionality to automatically ingest the details from a CV so there is less form filling for the user.

ANNEX A –Knowledge Discovery and Data Mining Technical Details

Data Pre-processing Phase: Transposing Dataset Algorithm

Figure A1 depicted the flow chart of a python script that was implemented to transpose the original dataset. The steps below listed out the flow of the python script.

Step 1: Read in the original dataset CSV file

Step 2: Extract the main job title

Step 3: Check the job title of next row

i. If it is different from the main job title, append the current job title into a list and jump to step 4.
ii. Otherwise, write the list of job titles into a new CSV file that contained the transposed dataset.

Step 4: Check if current row is the last row of record

iii. If yes, end the script.
iv. Otherwise, move to next row of record and repeat step 3.

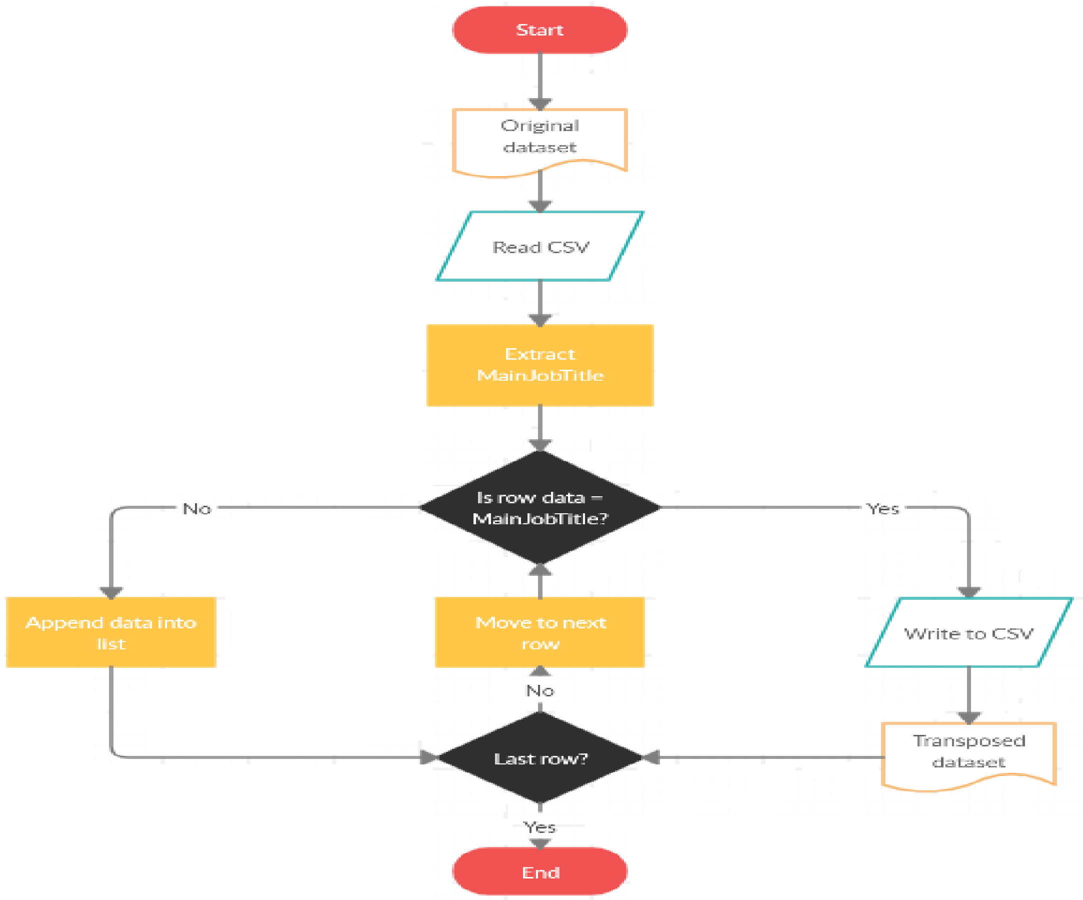


Figure A1

Knowledge Discovery Phase: Apriori Algorithm

Figure A2 depicted the flow chart of a python script that was implemented to identify item set that has no association. The steps below listed out the flow of the python script.

Step 1: Read in the transposed dataset CSV file

Step 2: Apply Apriori algorithm on the dataset

Step 3: Based on the preconfigured support, confidence and lift values, identify itemset that has no association and write it into a new CSV file

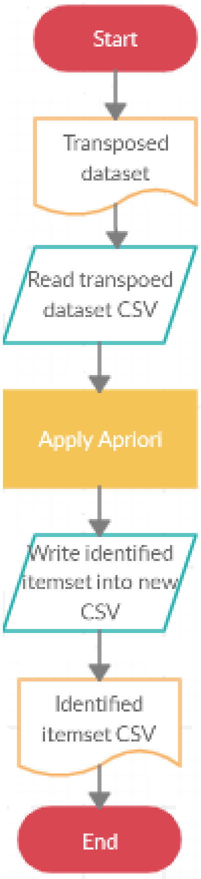


Figure A2

Data Post-processing Phase: Computation of Node to Node and Heuristic Cost Algorithm

Figure A3 depicted the flow chart of a python script that was implemented to calculate the node to node and heuristic cost. The steps below listed out the flow of the python script.

Step 1: Read in the post-processed dataset CSV file

Step 2: Create 2 lists: Node to node pair list and combined node to node and cost list

Step 3: Compute the sum of all common node to node pair and average cost of it

Step 4: Generate node to node cost CSV and append the average cost of node to node pair

Step 5: Compute the sum of all CV cost and average cost of it

Step 6: Generate heuristic cost CSV and append the average of total CV cost

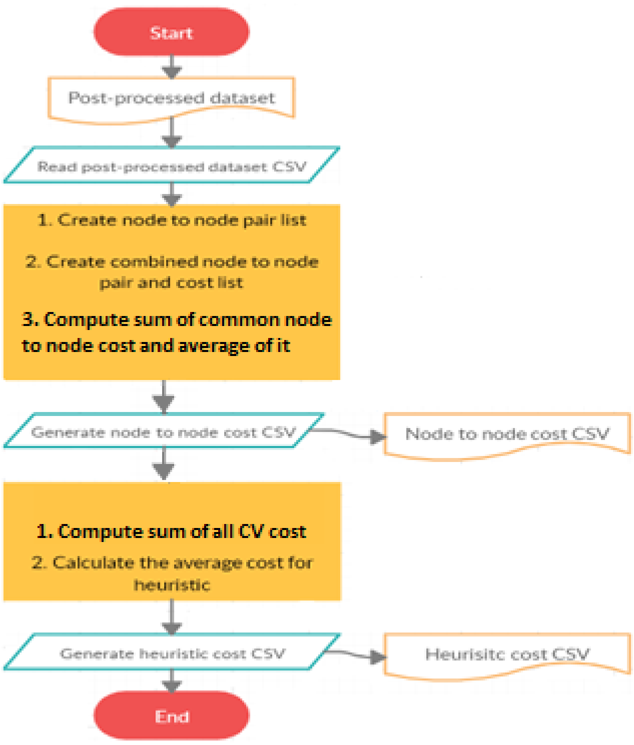


Figure A3

ANNEX B – List of Job Titles

Analyst
CIO
Consultant
COO
CTO
Director
Engineer
Head
Intern
Manager
Operation Manager
Presales Consultant
President
Principal Engineer
Principal Software Engineer
Product Manager
Professor
Programme Manager
Project Manager
Sales Director
Sales Manager
Senior Analyst
Senior Consultant
Senior Director
Senior Engineer
Senior Head
Senior Manager
Senior Presales Consultant
Senior Project Manager
Senior Sales Director
Senior Sales Manager
Senior Software Developer
Senior Software Director
Senior Software Engineer
Senior Software Manager
Senior Solution Architect
Senior Technical Manager
Software Developer
Software Director
Software Engineer
Software Manager
Solution Architect
Technical Lead
Technical Manager
Vice President

Annex C – Software engineer job competency

Competency	Frequency	Column1	Competency2	Frequency 3
SOFTWARE DEVELOPMENT	192		QA	23
WEB	184		HTML5	22
C	172		POST	21
JAVA	130		DEVELOPER	21
.NET	89		ENHANCEMENT	20
SQL	84		FAST	20
C++	84		DOCKER	20
UP	83		QUALITY ASSURANCE	20
EA	76		SERVERS	19
HARDWARE	72		BUG	19
DATABASE	65		ANALYTICS	19
INTERFACE	64		SYSTEM DESIGN	19
AUTOMATION	58		J2EE	17
LINUX	57		STACK	17
JAVASCRIPT	54		DEVICE	17
CLOUD	48		.NET FRAMEWORK	17
SDLC	41		APP	16
ASP.NET	37		KUBERNETES	16
API	35		JQUERY	16
SOFTWARE ENGINEERING	35		DEVOPS	16
SCRUM	34		IOT	15
PYTHON	33		UNIX	15
NETWORK	32		PROTOCOL	15
OBJECT	31		WPF	15
MYSQL	30		JSON	15
CSS	29		ALGORITHM	15
TROUBLESHOOTING	29		EMBEDDED SOFTWARE	15
ANDROID	29		SOFTWARE TESTING	15
SYSTEM SOFTWARE	28		CONTINUOUS INTEGRATION	15
R	27		CHARGE	14
MICROSERVICES	27		DESKTOP	14
DEBUG	27		OPERATING SYSTEM	14
SIMULATION	27		SIGNAL	13
REST	26		OS	13
HTML	26		VB	13
DEBUGGING	26		UML	13
PHP	26		AGILE SOFTWARE DEVELOPMENT	13
AWS	26		MONGODB	12
IOS	26		COLLABORATION	12
PROJECT MANAGEMENT	26		SMART	12
TECHNICAL SUPPORT	25		SOAP	12
SOFTWARE ARCHITECTURE	25		FIX	12
DATABASES	24		CSS3	12
FIRMWARE	24		XML	12
OBJECT-ORIENTED	24		INTERNET	12
SQL SERVER	24		VB.NET	12
MVC	23		PC	12
GIT	23		OPEN SOURCE	12
COMPLIANCE	23		PUBLIC	11