DATA SCIENCE
Machine Learning

Career Map

Yuxin Zhao[1] | Jinyi Luo[2]

[1]Statistics, George Washington University,
Email: yuxin@gwu.edu

[2]Statistics, George Washington University,
Email: jinyi@gwu.edu

This article studies the relationship between Resumes and Job descriptions. It intends to develop a model that provides hiring and career suggestions for employers and applicants. The first part is to match the Resumes with the positions, and to rank the Resumes by selecting criteria of the positions. By applying string similarity method to remove duplicated value, we are able to condense a Career Map in order to let candidates know their possible next step in the job market.

KEYWORDS
NLP, Data Scientist, NaiveBayes, String Similarity, TF-IDF, N-Grams, Cosine similarity, Neural Network, Word2Vec, Network Analysis

1 | BACKGROUND

During the recruitment procedure, applicants are searching for companies and companies are selecting applicants. Both wasted too much time on information filtering. Studies show that the high job-search cost and imperfect information would deteriorate the welfare of job-seekers, and the efficiency of labor market.

From the Resume dataset, we have thousands of millions of applicants listed all their career paths. Is that possible that we can use the methods of Natural Language Processing (NLP) and network analysis to generate a career map, that can provide suggestions to students?

"The Sexiest Job of the 21st Century" is awarded to Data scientist (Davenport et al. 2012), who are primarily tasked with taking raw data and using programming, visualization and statistical modeling to extract insights (Flowers , 2019). Therefore, data scientist and its related positions would be used as the main illustration in this article.

2 | PURPOSE

The purpose of this project is to use NLP methods to analyze the dataset of the Resumes and job descriptions. Develop a model that provides employers and applicants with hiring and career suggestions. The first part is to match the Resumes and the positions, and also to rank the Resumes by the positions' selecting criteria. The second part is using the methods of NLP and network analysis to generate a career map, that can provide suggestions to students. Finally we might improve the efficiency of a labor market by using these two models.

3 | DATA OVERVIEW

The dataset used in this article contained Indeed Job Descriptions and Resumes. In total there are 18,727 job posting which are posted by 9,658 companies. 560 Data Scientist positions are posted by 400 companies. Data Scientist is the rank 11 hottest job as well as rank 6 hottest job title. Among total job postings, the corresponding resumes are 3,564,157 for 28 jobs. As we can see the average job opening attracts 190 resumes, making the labor market very competitive.

Job	Number of Job Postings
Sales Manager	2171
Accountant	1756
Product Manager	1511
Marketing Manager	1351
Physician	1330
Financial Analyst	894
Software Developer	849
Social Worker	621
Paralegal	619
Construction Manager	581
Data Scientist	**560**
IT Manager	513
Mechanical Engineer	513
Web Developer	485
Management Analyst	384

Job Title	Number of Job Postings
Staff Accountant	247
Senior Accountant	188
Financial Analyst	187
Product Manager	184
Sales Manager	155
Data Scientist	**133**
Accountant	128
Software Developer	125
Marketing Manager	123
Product Marketing Manager	108
Senior Financial Analyst	90
Technical Writer	86
Web Developer	80
Construction Project Manager	80
Paralegal	78

FIGURE 1 Top 15 Job vs Job Title

Since our data set is large, we decide to use cloud computing to speed up the computation process. And we choose to use Amazon Web Services (AWS) by setting up AWS Management Console, and select Ubuntu Server 18.04 as the T2 Instances. Besides, we used PuTTY and FileZilla to connect to the AWS server and upload files.

For demonstration Purpose, we selected four job titles' Resumes located at DC area which are Data Scientist, Database Administrator, Computer System Administrator and IT manager. For each job title and location, there are thousands of resumes and tens of position descriptions. The Resume data are structured as dictionaries, while contents of job

descriptions are natural language.

TABLE 1 Titles Information

	All Resumes	All Titles	DC Resumes	DC Titles	DC Unique Titles
Database Administrator	53,613	351,697	5,503	100,970	18,348
Computer Systems Administrator	20,550	165,934	2,171	53,418	10,942
Data Scientist	18,467	104,782	1,298	27,181	5,220
IT Manager	257,602	1,749,375	19,181	137,042	77,061

Regarding Resume dataset, we found that most of the resumes contain only eight or less sections. Among all the 14 types of sections, experience, title, education, location, description, additional_info and skills are most likely to be used. As a result, in this project, we will mainly use experience and skills variables.

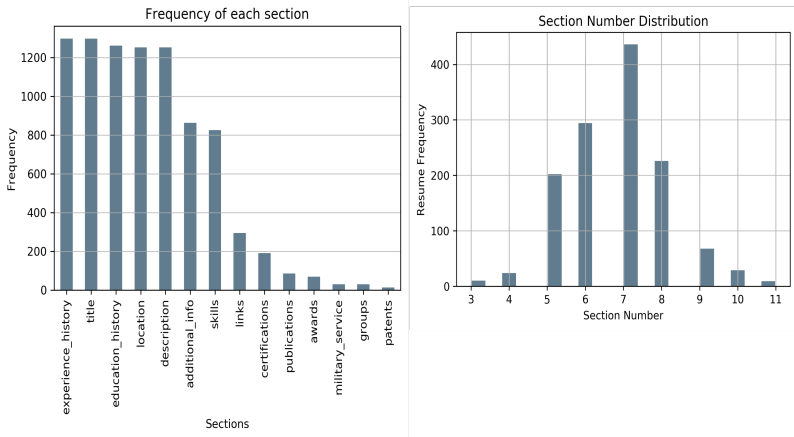


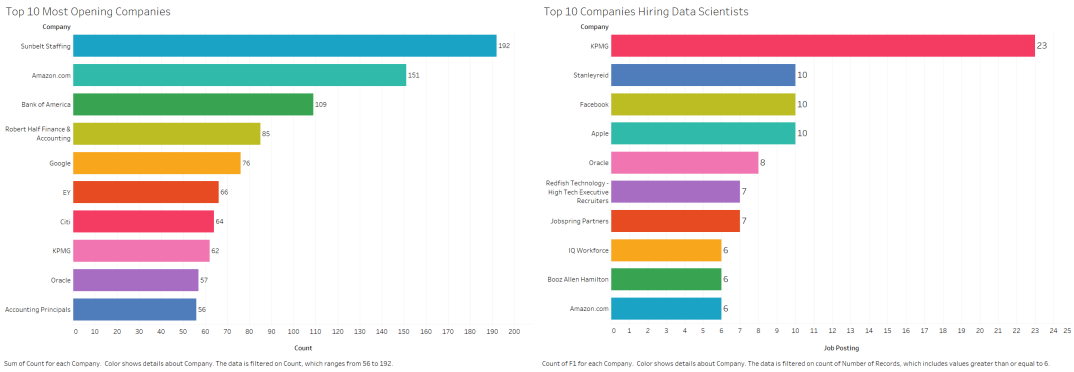
FIGURE 2 Resume Section Distribution

4 | EXPLORATORY DATA ANALYSIS

Before we build the model, we would like to explore the job description files, and answer some frequently asked questions that job seekers might have.

4.1 | What company has most openings?

As we can see from Figure 3, Sunbelt Staffing, Amazon, and Bank of America are the top 3 companies that had the most openings. While KPMG, Stanley Reid, Facebook, and Apple are the top hiring companies for Data Scientist.



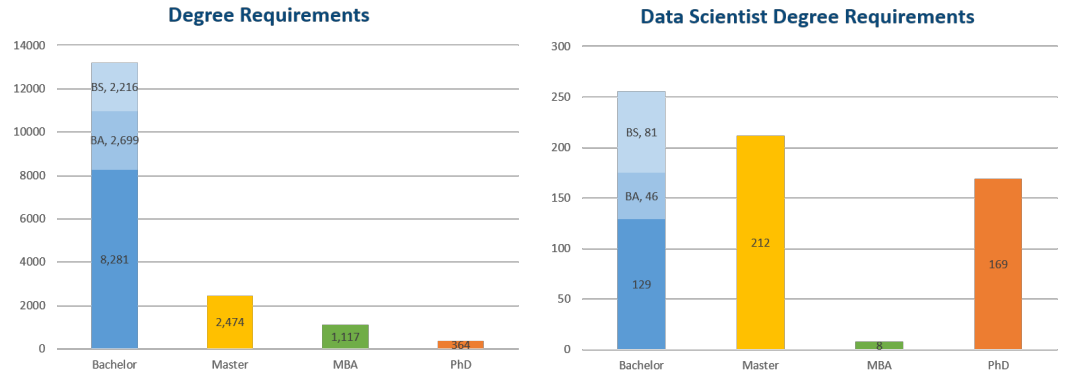
(a) Top 10 Most Opening Companies

(b) Top 10 Companies Hiring Data Scientists

FIGURE 3 Top 10 Companies Hiring for General Positions vs Data Scientists

4.2 | What education do you need?

From Figure 4, we can see that a bachelor degree might meet the requirements for general positions. However, Data Scientist position requires a higher education, and prefers the PHD degree.



(a) Degree Requirements

(b) Data Scientist Degree Requirements

FIGURE 4 Degree Requirements for General Positions vs Data Scientists

4.3 | What experience and skills do you need?

We are curious about what are the most significant experiences, education and skills in different job groups? Word cloud can help with that. As to experience history. Seems that analyst experience is also important for data scientists.

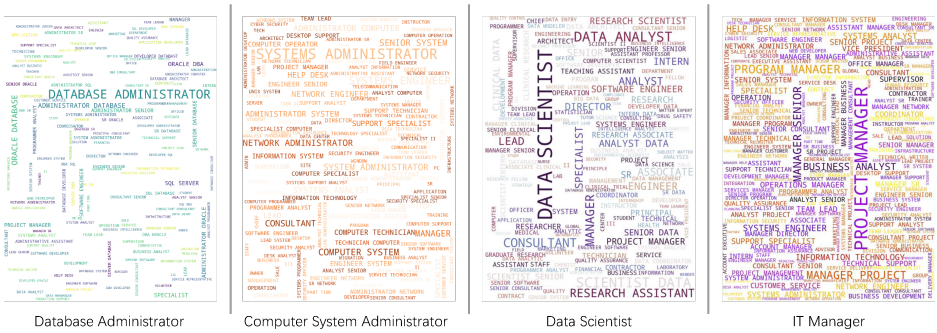


FIGURE 5 Word Cloud for Experience History

Skills is the most differentiate variable. Python and machine learning take the domain part of data scientist. Database and SQL are so important for database administrator. For computer system administrator, there are some skills I don't even know the meaning so well, such as CISCO and active directory.

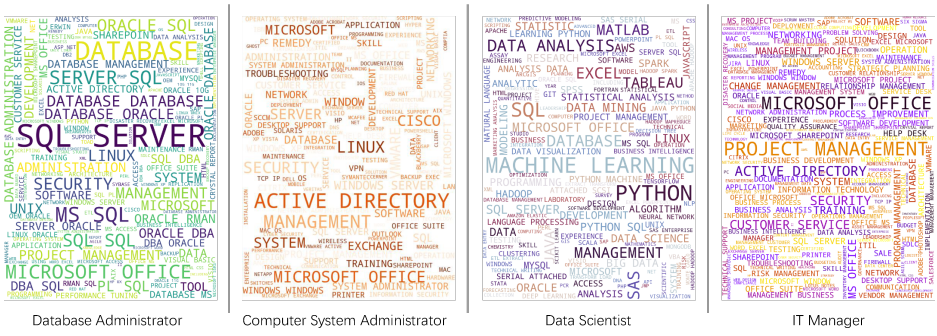
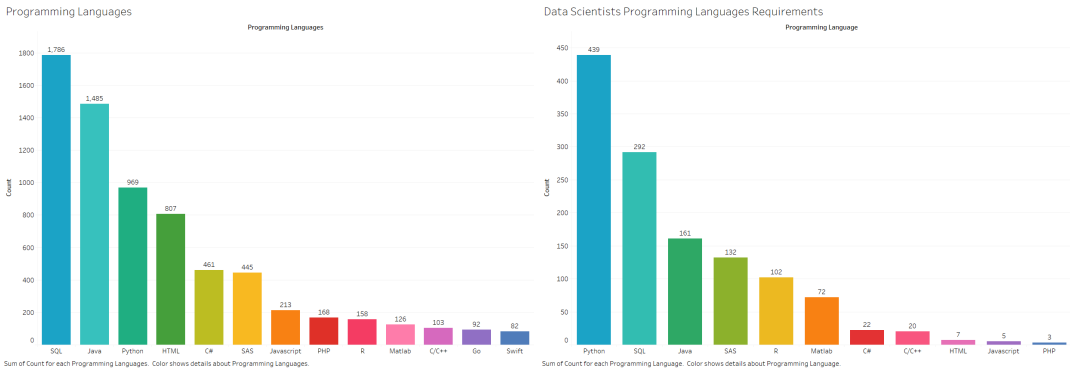


FIGURE 6 Word Cloud for Skills

From Figure 7, for the general positions, 6,895 Programming Skills are listed. Regrading Visualization Tools: 1,331 positions requires Google Analytics, 346 positions requires Tableau. For Statistical Modeling: 7,655 positions requires Excel, 2,881 positions requires R, and 431 positions requires SAS.

For the Data Scientists positions, 1,255 Programming Skills are listed. For Visualization Tools: 89 positions requires Tableau, 19 positions requires Google Analytics. For Statistical Modeling: 208 positions requires Excel, and 132 positions requires SAS.

As a result, seems that experience and skills are good variables that differentiated between different job titles. So we can use these two variables to build classification models to group the new Resumes.



(a) Programming Languages

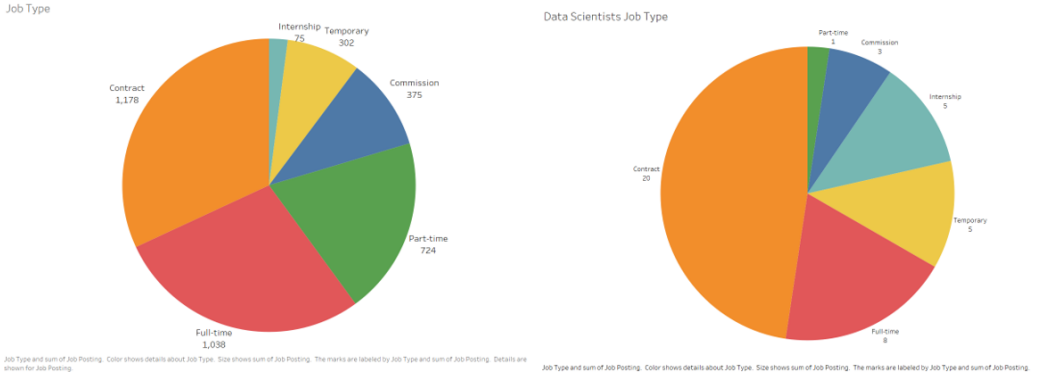
(b) Data Scientist Programming Languages

FIGURE 7 Programming Languages Requirements for General Positions vs Data Scientists

4.4 | What salary to expect?

4.4.1 | Job Type

From 2,421 Job Postings which contained Job Type, we can see that contract has the highest demand, and followed by the full-time positions. The internship occupied a relative small amount.



(a) Job Type

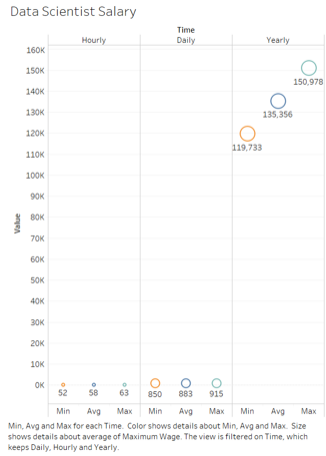
(b) Data Scientist Job Type

FIGURE 8 Job Type for General Positions vs Data Scientists

4.4.2 | Salary

From 2,312 Job Postings which contained Salary Info, we can indicate that the Data Scientist position has relatively higher salary compare to the general market.

(a) Salary



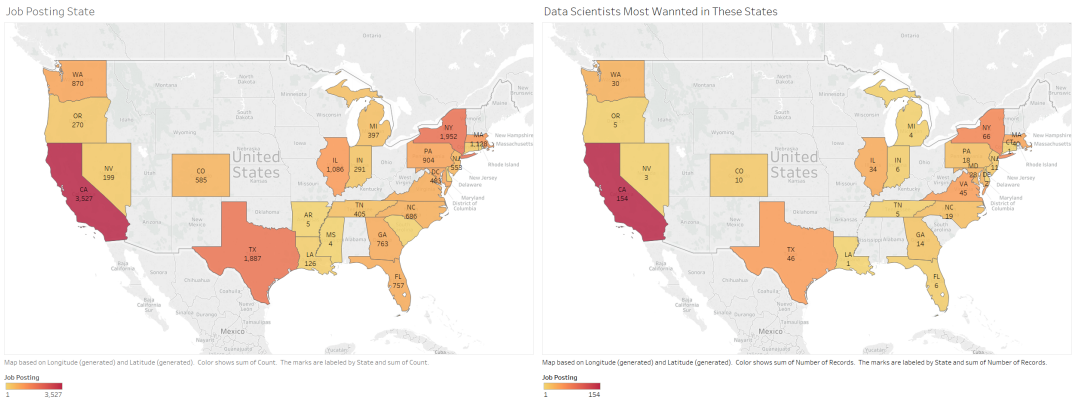
(b) Data Scientist Salary

FIGURE 9 Salary for General Positions vs Data Scientists

4.5 | Where to apply?

4.5.1 | Location - States

Unsurprisingly, New York and California are the top 2 hiring states for general market as well as specifically for data scientists positions. Besides, California pays the most especially for senior leveled data scientists positions.



(a) Top hiring states

(b) Top hiring states for Data Scientists

FIGURE 10 Top hiring states for General Positions vs Data Scientists

TABLE 2 The Salary of Data Scientist the Top 10 Hiring States

State	Senior Salary	Non-Senior Salary	Difference
CA	**167,000**	144,200	22,800
NY	165,000	129,167	35,833
MA	160,000	102,125	57,875
VA	155,000		
WA	144,000	163,250	(19,250)
IL	128,250		
PA	125,000	133,750	(8,750)
TX	120,000	125,000	(5,000)
CO	117,000	108,000	9,000
NJ	112,500		
DC		160,000	
NC		121,500	
IN		120,000	
MD		112,250	
DE		85,000	

4.5.2 | Location - Cities

From Figure 11, we can see that New York city has the most job postings for general market as well as for data scientists positions.

City	Number of Job Posting
New York	**1245**
San Francisco	751
Chicago	723
Houston	562
Atlanta	540
Los Angeles	522
Boston	517
Washington	479
Seattle	472
Dallas	416

Top hiring Cities for General Positions

City	Number of Job Posting
New York	61
San Francisco	54
Chicago	30
Seattle	24
Boston	23
Austin	18
Redwood City	15
Atlanta	14
Dallas	12
Washington	12

Top hiring Cities for Data Scientists Positions

FIGURE 11 Top hiring Cities for General Positions vs Data Scientists

4.6 | When to apply?

There are 9,753 jobs (52%) posted during 08/16 – 9/15/2018. Figure 12 represented the weekday distribution of those postings, and we could conclude that most of the jobs posted during Wednesday to Friday.

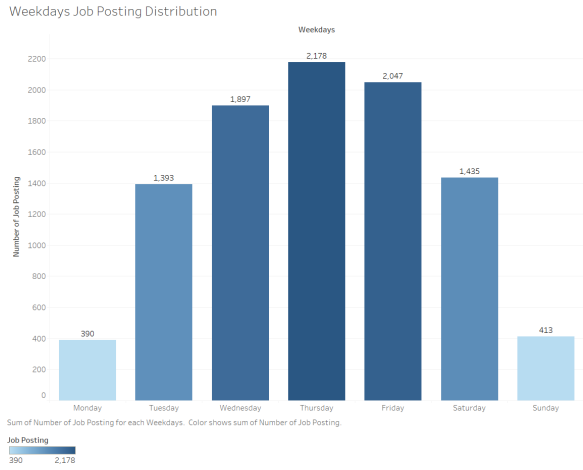


FIGURE 12 Weekdays Job Posting Distribution

5 | METHOD

5.1 | Naive Bayes Classifier

Naive Bayes is a classifier for machine learning that is simple but effective and commonly used. It is a probabilistic classifier that uses the Maximum A Posteriori decision rule in a Bayesian setting to make classifications.

Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \ldots, x_n)$ representing some n characteristics (independent variables), for each of K possible outcomes or classes C_k, it assigns to this instance probabilities $p(C_k \mid x_1, \ldots, x_n)$.

Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

The naive Bayes classifier combines this model with a decision rule. One common rule is to select the most likely hypothesis, this is called the maximum a posteriori or MAP decision rule.

5.2 | Fast String Matching

5.2.1 | TF-IDF

TF-IDF stands for Term Frequency-Inverse Document Frequency. Term Frequency (TF) measures how often in a document a term occurs. Due to every document is different in length, a term may appear in longer documents much more than the shorter ones. In order to normalize it, the term frequency is often divided by the document length, and the formula is as follows:

$$\text{TF(t)} = \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}} \tag{1}$$

Inverse Document Frequency (IDF) measures how important a term is, weighs down frequent terms while rare ones are scale-up, and the formula is as follows:

$$\text{IDF(t)} = \log \frac{\text{Total number of documents}}{\text{Number of documents with term t in it}} \tag{2}$$

Therefore, the formula of Term Frequency-Inverse Document Frequency is as follows:

$$\text{TF-IDF(t)} = \text{TF(t)} * \text{IDF(t)} \tag{3}$$

5.2.2 | N-Grams

N-grams are sequences of N contiguous items from a given sample of text. In our case, for example, the n-grams of the word "scientist", when n=3 becomes: ['sci', 'cie', 'ien', 'ent', 'nti', 'tis', 'ist']. While n-grams of the word "science", when n=3 becomes: ['sci', 'cie', 'ien', 'enc', 'nce'].

And we can then compute the TF-IDF matrix for n-grams of the word "scientist" as follows:

```
(0, 5053)    0.4009
(0, 1469)    0.3999
(0, 2869)    0.3912
(0, 2076)    0.3043
(0, 4014)    0.4103
(0, 5441)    0.4108
(0, 3045)    0.3108
```

The TF-IDF matrix for n-grams of the word "science" is as follows:

```
(0, 1469)    0.4157
(0, 2869)    0.4067
(0, 2064)    0.5236
(0, 3805)    0.4624
```

5.2.3 | Cosine Similarity

In order to compute the similarity between two vector of TF-IDF values, we would like to use cosine similarity, which will generate a metric that sayd how related are two documents by looking at the angle instead of magnitude.

$$\text{Cosine Similarity} = cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} \tag{4}$$

In order to compute the dot product, we utilized Compressed Sparse Row (CSR), which could more efficiently represent

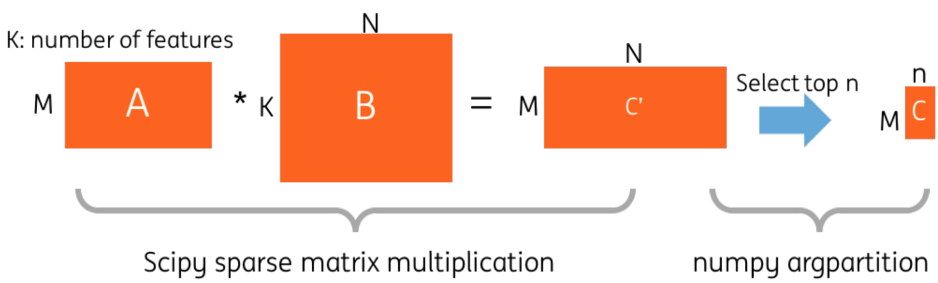


FIGURE 13 Use "sparse_dot_topn" to do sparse matrix multiplication and select the top-n entries

the sparse matrix, where most elements are zero.

We denote the following vectors:
A vector: size NNZ (non-zero elements), which stores the non-zero elements of the matrix.
IA vector: size M+1, which stores the cumulative number of non-zero elements up to (not including) the i-th row.
JA vector: size NNZ, which stores the column index of each element in the A vector.
Here is the Algorithm for representing the SPARSIFY (MATRIX) ("Sparse Matrix Representations", 2018):

Step 1: Set M to number of rows in MATRIX (5)

Step 2: Set N to number of columns in MATRIX (6)

Step 3: I = 0, NNZ = 0. Declare A, JA, and IA. Set IA[0] to 0 (7)

Step 4: for I = 0 ... N-1 (8)

Step 5: for J = 0 ... N-1 (9)

    If MATRIX [I][J] is not zero (10)

    Add MATRIX[I][J] to A (11)

    Add J to JA (12)

    NNZ = NNZ + 1 (13)

    [End of IF] (14)

Step 6: Add NNZ to IA (15)

    [End of J loop] (16)

    [End of I loop] (17)

Step 7: Print vectors A, IA, JA (18)

Step 8: END (19)

However, scipy sparse matrix multiplication is very costly, which need to compute the M X N matrix. To reduce the computation as well as save memory, we used "sparse_dot_topn" function, which provide a fast way to performing a sparse matrix multiplication and select top-n cosine similarity score (WB Advanced Analytics, 2017).

5.3 | Word2vec

Word2vec is a neural network that processes text in two layers. Its input is a corpus of text and its output is a set of vectors: feature vectors for words in that corpus. Although Word2vec is not a deep neural network, it turns text into a numerical form that can be understood by deep nets.

Word2vec's purpose and utility is to group the vectors of similar words together in vectorspace. That is, it detects mathematical similarities. Word2vec creates vectors, such as the context of individual words, that are distributed numerical representations of word characteristics. Without human intervention, it does so.

6 | DATA PRE-PROCESSING

First of all, we did some preprocessing regarding all the job titles from candidate's experience history. For example, we have got 18,467 data scientist resumes from 30 cities among 20 states. Before preprocessing, there are 104,782 titles, and 27,181 unique titles. After we remove all the abnormal symbols, and split the titles by "/", "&", "-", we got 47,123 flatted title list, and total unique titles became 23,835.

When we initially put all the applicants' paths in to the software Gephi, it gave us a network plot with too much noises. Cause some of the title node is linked to several title nodes with same meaning but different expression. So the first step is to identify the different expressions of the same meanings. It not only helps with the career map, but also can help with the Resume-position matching system. Here we came up with two methods to estimate the similarities between words. One is Fast String Matching, and another one is Word2vec.

6.1 | Fast String Matching

In our case, we stores the top 100 most similar items, and only show items with similiarity above 0.85. By using Cosine Similarity, we are able to detect the difference between administrator vs administration, administrator vs admin, sr vs senior, science vs sciences, intern vs internship. Besides, we can also detect the similar words but in different order, typo like "scienist", and remove duplicate like administrator administration.

379	computer systems administrator	computer systems administration	0.928745
383	computer systems administrator	computer systems administrator administration	0.901231
389	computer systems administrator	computer systems admin	0.858994
2	senior systems administrator	systems administrator senior	0.959432
12	senior systems administrator	senior systems admin	0.857253

FIGURE 14 Computer Systems Adminstrator

2638	sr technical support engineer	senior technical support engineer	0.865084
4514	senior network systems administrator	sr network systems administrator	0.87678
4972	senior technical support engineer	sr technical support engineer	0.865084

FIGURE 15 Sr versus Senior

139	data science	data sciences	0.927687
8	data scientist	d data scientist	0.86019
692	data scientist intern	d data scientist intern	0.910315
1104	data scientist summer intern	data scientist summer internship	0.864113
500	senior data scientist	sr senior data scientist	0.858319
501	senior data scientist	senior data scienist	0.856755

FIGURE 16 Data Scientist

6.2 | Word2Vec

Firstly we tried to use career path to train the model, which means the previous title as the input and the following title as the label, but it turns out to give a bad performance on words similarities. Then we decided to follow the skip-gram method, just using the neighborhood words as the input word's label.

For example, for the title "senior investment analyst financial sector". The first pair of input will be "senior", and its label will be "investment". And then, the word "investment" becomes the next input. It will generate two pairs of data because it has two neighbor words. One label is "senior" and another is "analyst". And so on.

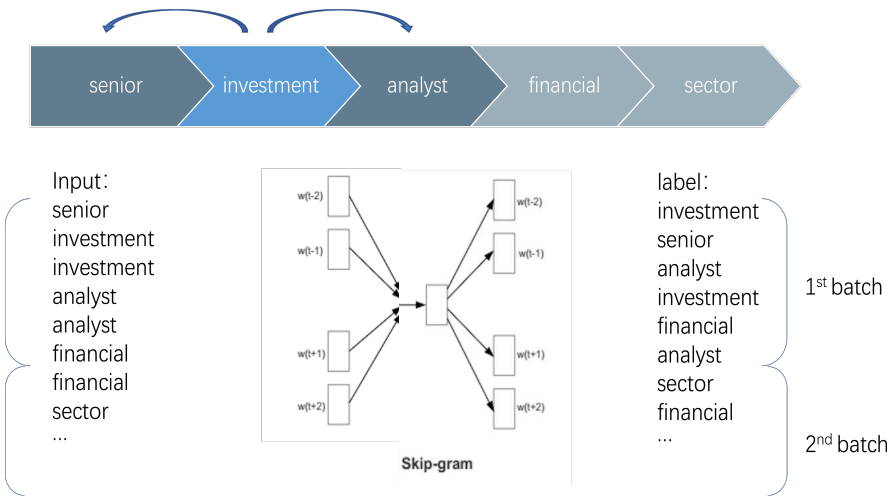


FIGURE 17 Word2vec Illustration

Then we need to decide the batch size of training, here we decided to put 8 pairs of elements into one batch. Now we have the batches and labels that we can train the data with a 1 hidden layer neural network to get the weight matrix as

each words' neural embedding.

Here's the nearest words outcome after 16000 iterations. As a result, the word2vec method gives a nice words similarity estimate.

Nearest to manager:	coordinator, administrator, specialist, manger, managementit, gems, midas, gelco,
Nearest to it:	cm, technology, bonds, reclamation, mis, initiatives, dedicated, publication,
Nearest to administrator:	admin, administration, manager, hatts, engineer, adminstrator, analyst, mariadb,
Nearest to senior:	sr, junior, jr, rac, aministrator, aip, apps, dessert,
Nearest to analyst:	engineer, administrator, booking, analysts, juniour, standardization, midrange, analysis,
Nearest to project:	program, shift, tests, directive, kitchen, store, asset, module,
Nearest to systems:	system, assuranceengineer, cnd, security, adv, isim, 13, netbackup,
Nearest to engineer:	administrator, engineering, architect, analyst, staas, technician, earned, specialist,
Nearest to consultant:	apps, ecg, engineerir, contractor, specialist, caregiver, operations–, cryogenic,
Nearest to specialist:	coordinator, manager, mariadb, edw, administrator, epr, officer, engineer,
Nearest to database:	dba, discoverer, subcontract, hpov, apr, plsql, antivirus, internships,
Nearest to support:	publishing, appliance, ec, atm, partitioned, checkout, databasearchitect, help,
Nearest to lead:	leader, practices, policing, reconstruction, tse, platformreporting, 204, status,
Nearest to sr:	senior, jr, 2013, server, aministrator, native, server2008, etf,
Nearest to director:	cafe, vice, aspects, commissioner, g3, editor, interactive, manager,
Nearest to assistant:	asst, connecticut, admitted, tunnel, associate, hopital, reform, retinal,

FIGURE 18 Nearest Words generated by word2vec

As a result, for "manager", it can distinguish the typo; manger, synonym: coordinator, administrator, managementit, and also the abbreviation: mgr sometimes. For "analyst", it can distinguish not only plural: analysts, but also analysis. For "administrator", it sloves admin, administration, manager and adminstrator.

The outcomes are reasonable and they can give us some information. But there are also something wrong here. For word "senior", "junior" will be the second closest word, because they have similar location and similar accompany words, but they have totally different meaning.

In a conclusion, we cannot simply apply this list to clean the resume title. Here the Jaccard coefficient similarity is useful again. Word2vec and Jaccard similarity, one can identify similar location, one can identify similar shape. If two words have both similar shape and similar meaning and location, they are safe enough to be replaced.

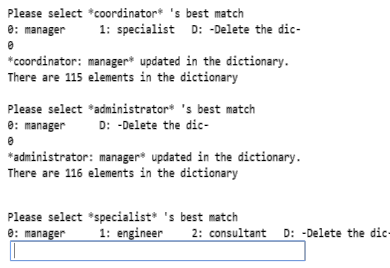


FIGURE 19 Manually Decide Function

For those word with totally different shape but similar meaning, we also provided a manually decision approach to decide they are replaceable or not as Figure 19 shown.

Then a full replacement dictionary is generated. In our logic, a word can only be replaced by the words with higher frequency. The replacement dictionary will be applied before we generate the career map.

Original	Replaced by	Original	Replaced by	Original	Replaced by	Original	Replaced by	Original	Replaced by
manger	manager	solution	solutions	server2008	server	analyist	analyst	centner	center
coordinator	manager	applicatino	applications	administer	administrator	assistance	assistant	staffs	staff
adminstrator	administrator	director	manager	analysis	analyst	trainee	intern	implementations	applications
administration	administrator	supervisor	manager	engineerir	engineer	operational	application	informatics	communications
analysts	analyst	admin	administrator	developers	developer	administative	administrative	administratortier	administrator
system	systems	curator	administrator	computers	computer	analayst	analyst	programs	program
engineering	engineer	sr	senior	technologyengineer	technology	geodatabase	database	executuve	executive
databse	database	principal	senior	chiefs	chief	sytems	systems	adiministrator	administrator
applications	application	program	project	researcher	research	assistants	assistant	transmission	information
leader	lead	technician	specialist	communication	communications	netwrok	network	technologyofficer	technology
techhnical	technical	apps	applications	management	manager	adminstrative	administrative	assocaite	associate
11information	information	promoter	support	aministrator	administrator	assuranceengineer	assurance	officerseries	officer
officerinformation	information	vp	president	assistantship	assistant	analysist	analyst	leaders	leader
deveoper	developer	asst	assistant	mangement	management	databaase	database	technologies	technology
operation	operations	tech	technician	tehnology	technology	microcomputer	computer	science	scientist
managementit	manager	financial	business	offices	office	sysems	systems	contract	contractor
services	service	control	management	directorexecutive	executive	technologyspecialist	technology	coodinator	administrator
architecture	architect	mgt	management	servers	server	reseach	research	asistant	assistant
i	ii	arch	architect	assuranceanalyst	assurance	managment	management	sever	server
iii	ii	rep	representative	assuranceconsultant	assurance	managements	management	enginee	engineer
mssql	sql	chief	officer	trainor	contract	customs	customer	coordinaor	coordinator
accounts	account	internship	intern	design	designer	managementengineer	management	telemarketing	marketing
principle	principal	iv	ii	administrators	administrator	oracle7	oracle	contracted	contract

FIGURE 20 Replacement Dictionary

7 | RESUME-POSITION MATCHING

We have decided which part of data we can use. Here comes the model building part. To recommend the best matching applicants for each position, we need to rank the applicants' ability scores and fetching those top ones. But if we got millions of resumes in our database, comparing all the resumes for each position's criteria could be time consuming. It will be more efficiency if we classify all the applicants into different groups. In the next step, we need to decide which group of applicants does this position want. In the third step, we can simple rank the applicants in the target group by the position's selecting criteria, which is computational friendly. At last, we choose those top score applicants and recommend them to the company.

7.1 | Step1: Grouping

In the first step, we need to classify all the applicants into different groups. According to the word cloud, experience history and skills are good variables that we can use to classify the jobs. If we classify with skill variable, the accuracy will be 77%. Experience variable will improve the accuracy to 83%.

```
In [58]:  # Compute the accuracy
          from nltk.classify.util import accuracy
          accuracy(nb_classifier, test_feats)
          # 0.8328877005347594

Out[58]: 0.8328877005347594
```

FIGURE 21 Classification Accuracy

The list in 22 shows the most informative features while classifying. Skill R is highly significant for data scientist, and database administrator's most important skills are RMAN and OEM.

```
rman = True                dataAd : itMana = 444.9 : 1.0
spark = True               dataSc : itMana = 156.8 : 1.0
oem = True                 dataAd : itMana = 134.5 : 1.0
scala = True               dataSc : itMana = 119.4 : 1.0
natural = True             dataSc : itMana = 113.2 : 1.0
rac = True                 dataAd : itMana = 104.7 : 1.0
machine = True             dataSc : itMana = 94.0 : 1.0
hive = True                dataSc : itMana = 85.5 : 1.0
neural = True              dataSc : itMana = 76.3 : 1.0
algorithms = True          dataSc : itMana = 74.3 : 1.0
r = True                   dataSc : comput = 70.5 : 1.0
linear = True              dataSc : itMana = 65.5 : 1.0
predictive = True          dataSc : itMana = 64.4 : 1.0
hadoop = True              dataSc : itMana = 60.2 : 1.0
directory = True           comput : dataSc = 59.1 : 1.0
pcr = True                 dataSc : itMana = 57.8 : 1.0
tuning = True              dataAd : itMana = 56.7 : 1.0
cisco = True               comput : dataSc = 52.7 : 1.0
visualization = True       dataSc : itMana = 50.1 : 1.0
dba = True                 dataAd : itMana = 50.0 : 1.0
```

FIGURE 22 The Most Informative Classify Features

7.2 | Step2: Position Classification

As to step 2, we need to find the appropriate group for a position. Previously, we computed similarities between the job description and each group's resume data, and rank the similarities to choose the highest rating group. But we found that the Jaccard's coefficient is not differentiated and the edit distance is time consuming.

Then we found another trickier way to extract the key words of job description. Basically, in DC area, we have thousands of Resume data and only tens of job description data. Among those Resumes, there are thousands of applicants using their own words to describe their skills. Fortunately, our Resume data is well structured, and it is easy to extract all the skill names from the Skills variable. We can store them as a thesaurus.

To extract the key words from the job description content, we can simply collect all the tokenized words and N-gram phrases, then keep only those words and phrases existing in the skill thesaurus.

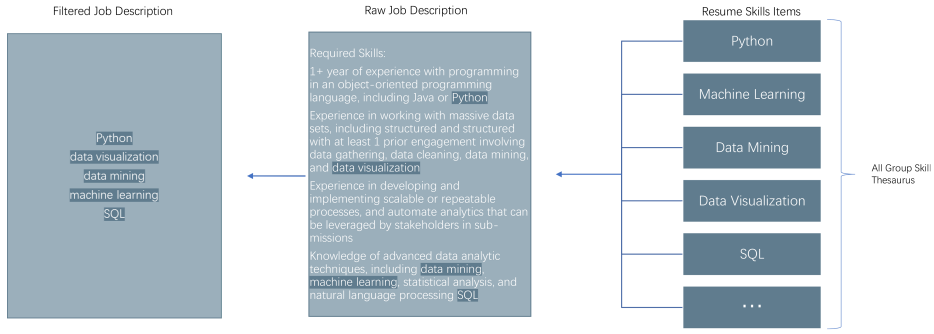


FIGURE 23 Target Skills of Positions

Then we can use these extracted key words to compare with the thesaurus of different groups by computing the frequency of each skill its own thesaurus. While the same job title could share some common skills. And we can compute the skill frequency in each group. Probably Python will have the highest frequency in data scientist group. We compute the percentage of each skills' frequency out of its group's total frequency. The percentage implies the importance level of this skill in its group. And then we compute the summation of all the target skills' frequency percentage in each group. The group with highest group score will be the target group of the next ranking step.

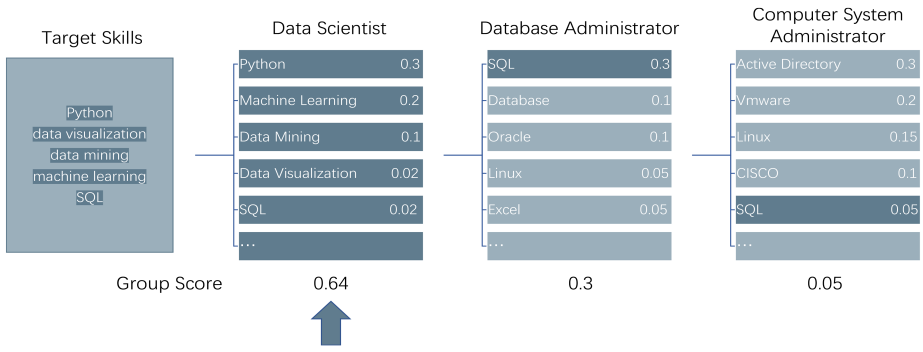


FIGURE 24 Position Classifying

7.3 | Step3: Scoring

After all the resumes and position can be classified properly, the next step is to rank all the applicants in the target group and recommend the top score applicants to the companies. We will keep using the key words extracted from each job description as the target skills. For each applicant in the target group, we add the total experience years of each target skills he has as his skill score.

For example, the Booz Allen Hamilton's data scientist position has target skills such as: Tableau and R and so on. The applicant number 76 listed his skills as followed. His skills matching this position are Data Analysis, Data Mining,

Statistical Analysis, Python and so on. The total experience years of these 7 skills is 61, that will be his skill score for this position. After we compute all the applicants' skill score in the target group. We can simply rank them to recommend the top several ones to the company.

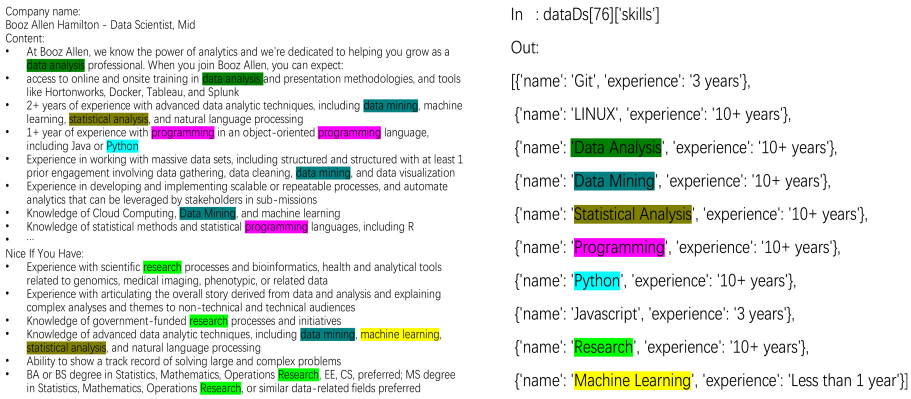


FIGURE 25 Skill Scoring Example

7.4 | Model Performance

These are the results of position classifications and applicants recommendations. When we choose the 11^{th} data scientist job description content, using at most 4-gram, the function will diagnose that Booz Allen Hamilton's position might belong to the Data Scientist group. Highly recommended applicants' skill scores for Booz Allen Hamilton's position form Data Scientist groups are listed as followed.

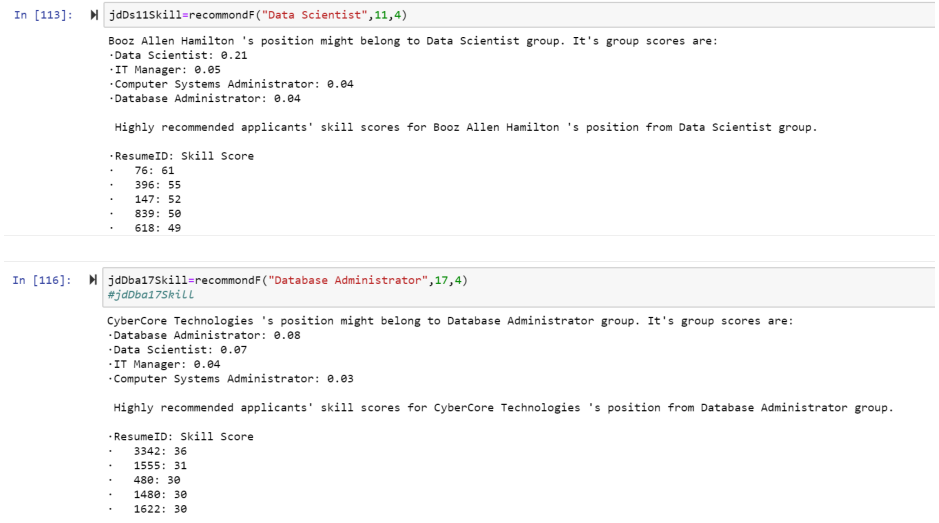


FIGURE 26 Resume Position Matching Outcomes

We want to check the correctness of the recommendation. Here's part of the 11^{th} data scientist job description content. And the 396^{th} data scientist resume with the highest skill score. Seems the recommendation is reasonable, the 396^{th} applicant have more than 10 years experience of several target skills.

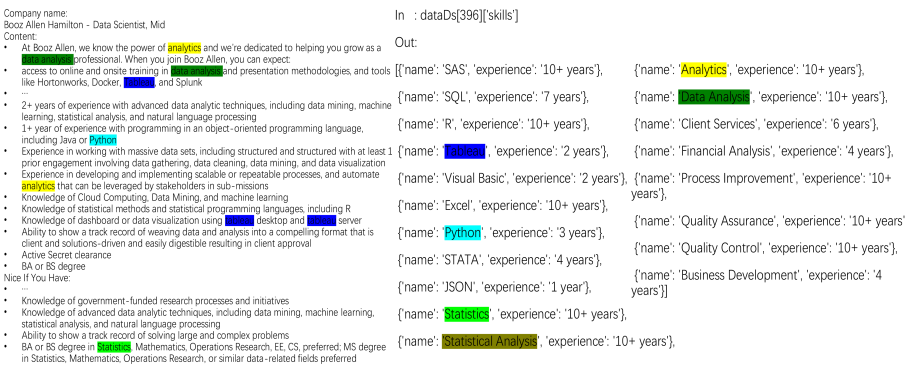


FIGURE 27 Matching Outcome Validation

When we choose a database administrator job description, it also classified correctly, and ranked the applicants' skill score in the database administrator group.

8 | CAREER MAP

After applied the replacements generated by string similarity methods, we can generate a career map with edge weight greater than 20, which means these paths have been repeated for at least 20 times in our resume dataset. It can keep only those general paths, instead of too specific details.

Here we have four job fields' paths. The four different colors stand for database administrator, data scientist, IT manager, and computer system administrator. As a result, in our career map, the most significant path is from data analyst to data scientist. The overall career map has a nice line shape, the left side and the right side are not directly connected. Some points are at the end, they are more likely to be in senior or manager positions.

There is a clear center line, nodes on this line have high centrality. Edges between those nodes are bridges among different fields. If applicants want to go into a new career field, these points are the necessary experiences they need to go through.

The career map can be applied to provide suggestions not only for applicants but also for companies. If an employer received an application from a database manager, he can recognize that this applicant is less likely to do a good job as a system administrator, because they are seldom linked together.

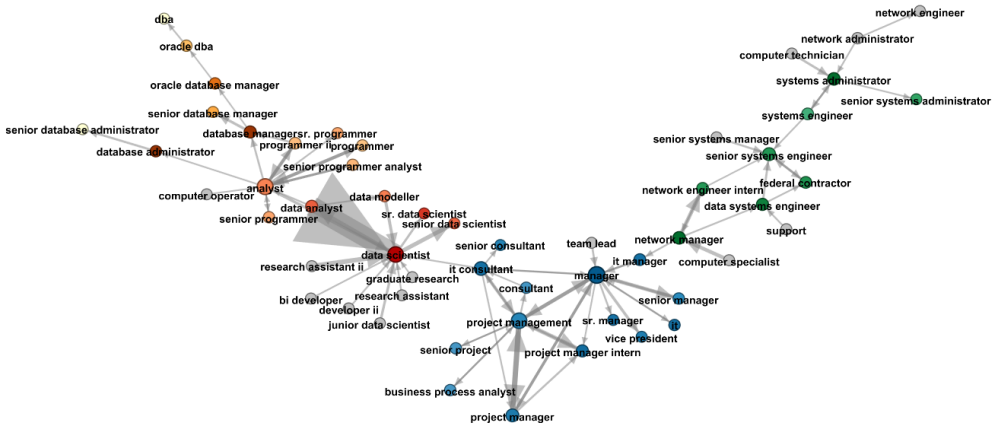


FIGURE 28 Career Map

For applicants, the career map is even more useful. We were meant to build an interactive function for applicants use. When an applicant uploads his resume, our system will quickly identify which stage is he at, and provides several possible career paths for him to choose. After he selected his dream position, we could tell him which skills he needs to improve urgently, and even recommend some course and books for him to learn.

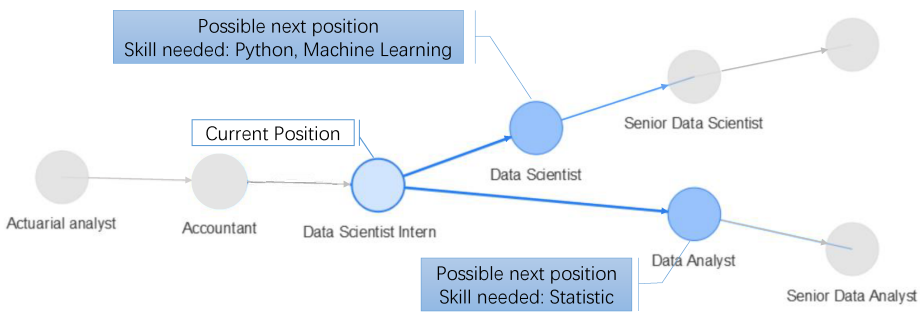


FIGURE 29 Career Map Application

REFERENCES

Davenport, T. H., & Patil, D. (2012, October). Data Scientist: The Sexiest Job of the 21st Century.
Retrieved from https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century

Den, V. (2017, October 14). Super Fast String Matching in Python.
Retrieved from https://bergvca.github.io/2017/10/14/super-fast-string-matching.html

Devin Soni. (2018, May 16). Introduction to Naive Bayes Classification.
Retrieved from https://towardsdatascience.com/introduction-to-naive-bayes-classification-4cffabb1ae54

Flowers, A. (2019, April 05). Data Scientist: A Hot Job That Pays Well.
Retrieved from https://www.hiringlab.org/2019/01/17/data-scientist-job-outlook/

Goel, A. (2019, March 18). 10 Best Programming Languages to Learn in 2019 (for Job & Future).

Perone, C. S. (n.d.). Machine Learning: Cosine Similarity for Vector Space Models (Part III).
Retrieved from http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/

Skymind. A Beginner's Guide to Word2Vec and Neural Word Embeddings.
Retrieved from https://skymind.ai/wiki/word2vec

Sparse Matrix Representations | Set 3 (CSR). (2018, January 03).
Retrieved from https://www.geeksforgeeks.org/sparse-matrix-representations-set-3-csr/

WB Advanced Analytics. (2017, July 26). Boosting selection of the most similar entities in large scale datasets.
Retrieved from https://medium.com/wbaa/https-medium-com-ingwbaa-boosting-selection-of-the-most-similar-entities-in-large-scale-datasets-450b3242e618

Wikipedia. (2019, April 30). Naive Bayes classifier.
Retrieved from https://en.wikipedia.org/wiki/Naive_Bayes_classifier