



How to find best fit candidates for a role using NLP

Bindu Jacob

Capstone project Report

SpringBoard Data Science Career track

1. Introduction

1.1 About

Companies are looking to hire personnel that are best fit for their job requirements from the vast number of applicants that post their resume on sites like LinkedIn. They want to get access to best aligned candidates, with just keyword searches like 'Aspiring human resources' or 'Seeking english teacher'.

1.2 Who Benefits?

Any company who would like to categorize their pool of resumes to find and aggregate best fit candidates from search strings thus bypassing manually allocating resumes for further evaluation.

The keyword strings used to access the best fit candidates can be succinct and improved to get accessed to best candidates for the job requirements.

1.3 Problem Statement

How to find best fit candidates for a role based on their available information from keyword searches

2. Data Wrangling

2.1 Data Overview

The data comes from sourcing efforts. Removed any field that could directly reveal personal details and gave a unique identifier for each candidate.

The data is contained in an excel sheet. It has 104 observations and 5 attributes. Attributes are Id, job_title, location, connection and fit. Only attributes job_title and location are considered for further text analysis

id	job_title	location	connection	fit
----	-----------	----------	------------	-----

	id	job_title	location	connection	fit
0	1.0	2019 C.T. Bauer College of Business Graduate (Magna Cum Laude) and aspiring Human Resources professional	Houston, Texas	85.0	NaN
1	2.0	Native English Teacher at EPIK (English Program in Korea)	Kanada	500+	NaN
2	3.0	Aspiring Human Resources Professional	Raleigh-Durham, North Carolina Area	44.0	NaN
3	4.0	People Development Coordinator at Ryan	Denton, Texas	500+	NaN
4	5.0	Advisory Board Member at Celal Bayar University	İzmir, Türkiye	500+	NaN

2.2 Data Cleaning

The data had no missing data.

About 50% of the data was duplicate and removing these resulted in 51 records.

As only locations from US and Canada were considered, 2 records that had location entry as Turkey was removed.

Final data contained 49 records.

2.3 Text Preprocessing

Text preprocessing is done on the job title and location text to make them more amenable for analysis, and for reliable results.

The texts job_title and location are combined into one column for further analysis.

To do preprocessing, regular expression was used to remove any punctuation, removal of commonly occurring words and then lowercase the text.

2.3.1 Remove punctuation and digits, stopwords and Tokenize

- Using nltk and re libraries, the punctuations and phone numbers are removed from the text.
- Stop words which are commonly used words like 'the', 'a', 'an', 'in' etc are removed.
- Each job_title and location texts are then tokenized

2.3.2 Location

Location attribute required the following cleaning:

- Canada was missplet as 'Kanada' in the original text, which was corrected.
- Words like 'area' , 'city' and 'greater' were removed from location
- Some of the places had missing states, these were manually included
- punctuations like '/', '-' and whitespaces were removed

	job_title	location	connection	job_title_nostop	location_cleaned	combined
0	2019 C.T. Bauer College of Business Graduate (Magna Cum Laude) and aspiring Human Resources professional	houston, texas	85.0	2019 ct bauer college business graduate magna cum laude aspiring human resources professional	houston texas	2019 ct bauer college business graduate magna cum laude aspiring human resources professional houston texas
1	Native English Teacher at EPIK (English Program in Korea)	canada, canada	500.0	native english teacher epik english program korea	canada canada	native english teacher epik english program korea canada canada
2	Aspiring Human Resources Professional	raleigh durham, north carolina	44.0	aspiring human resources professional	raleigh durham north carolina	aspiring human resources professional raleigh durham north carolina

	job_title	location	connection	job_title_nostop	location_cleaned	combined
3	People Development Coordinator at Ryan	denton, texas	500.0	people development coordinator ryan	denton texas	people development coordinator ryan denton texas
4	Aspiring Human Resources Specialist	new york, new york	1.0	aspiring human resources specialist	new york new york	aspiring human resources specialist new york new york

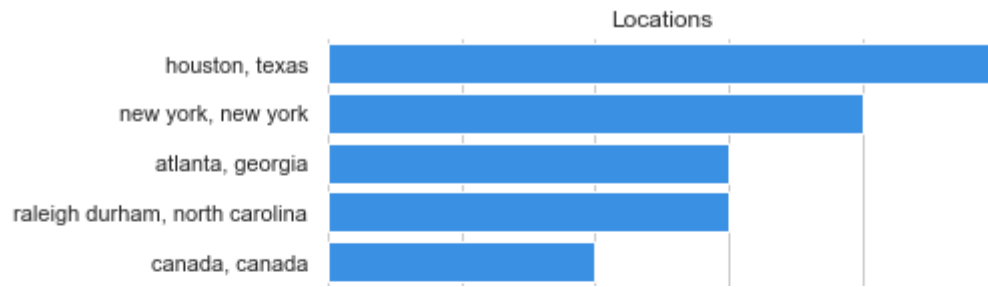
3. Exploratory Data Analysis

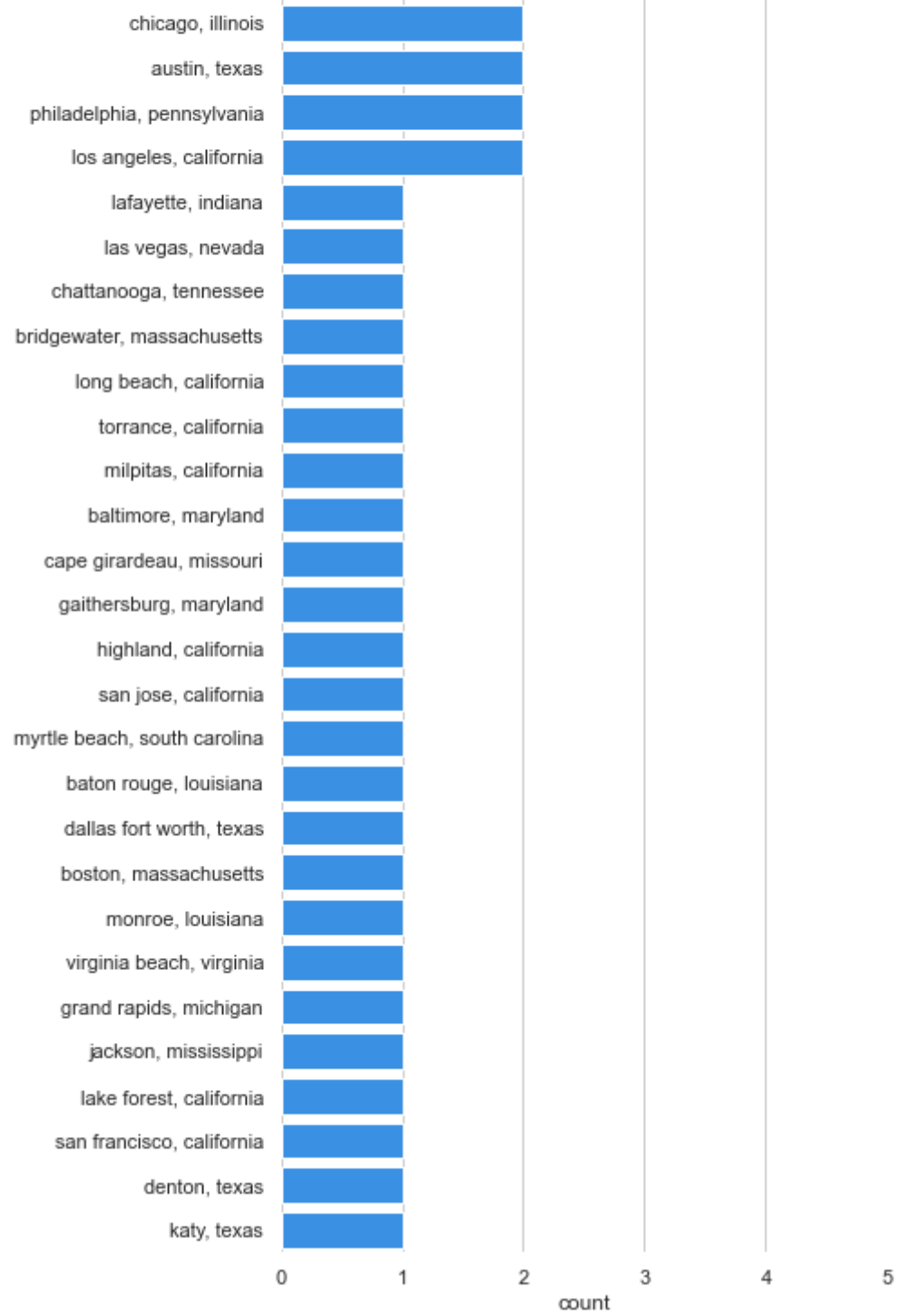
3.1 Investigation

For further analysis the job titles and location text were explored here to understand the data

3.1.1 Analyzing Location text

The data analysis of location shows that Houston, Texas has the most number of candidates, closely followed by New York and Atlanta.





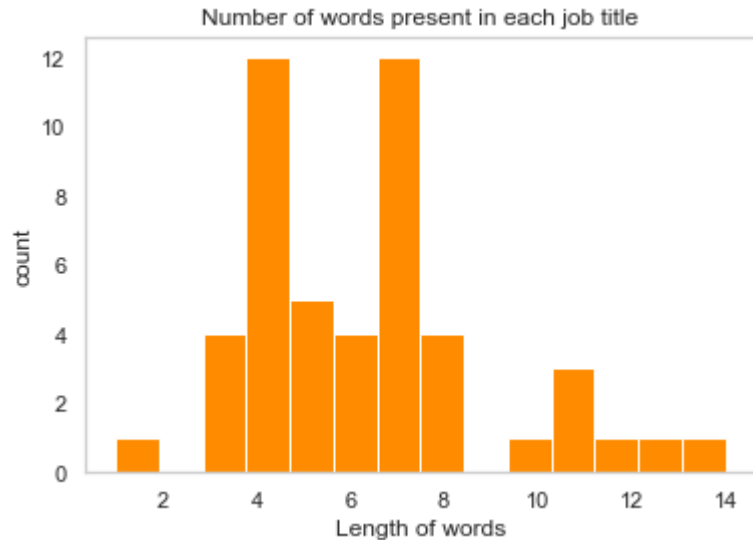
3.1.2 Analyzing job title text

To verify the preprocessing, a word cloud using the wordcloud package is plotted to get a visual representation of most common words.

It is key to understanding the data and ensuring we are on the right track, and if any more preprocessing is necessary before training the model.



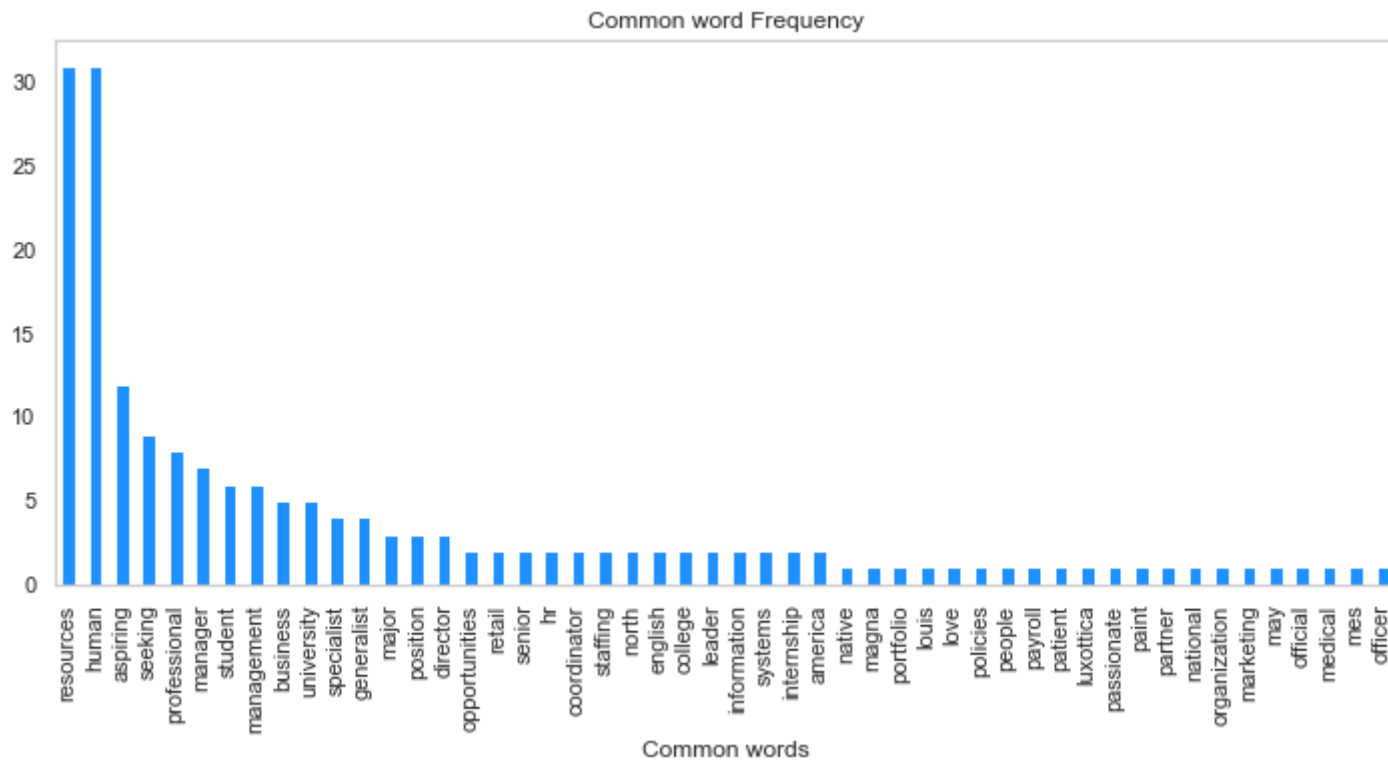
The plot shows that job title contains words with lengths 1 to 14 and average word length is between 4 to 7. We can see one outlier with word length 1



3.1.3 Term frequencies

A document term matrix is created with CountVectorizer and plotted to find the most frequent data found in the entire text of the job titles.

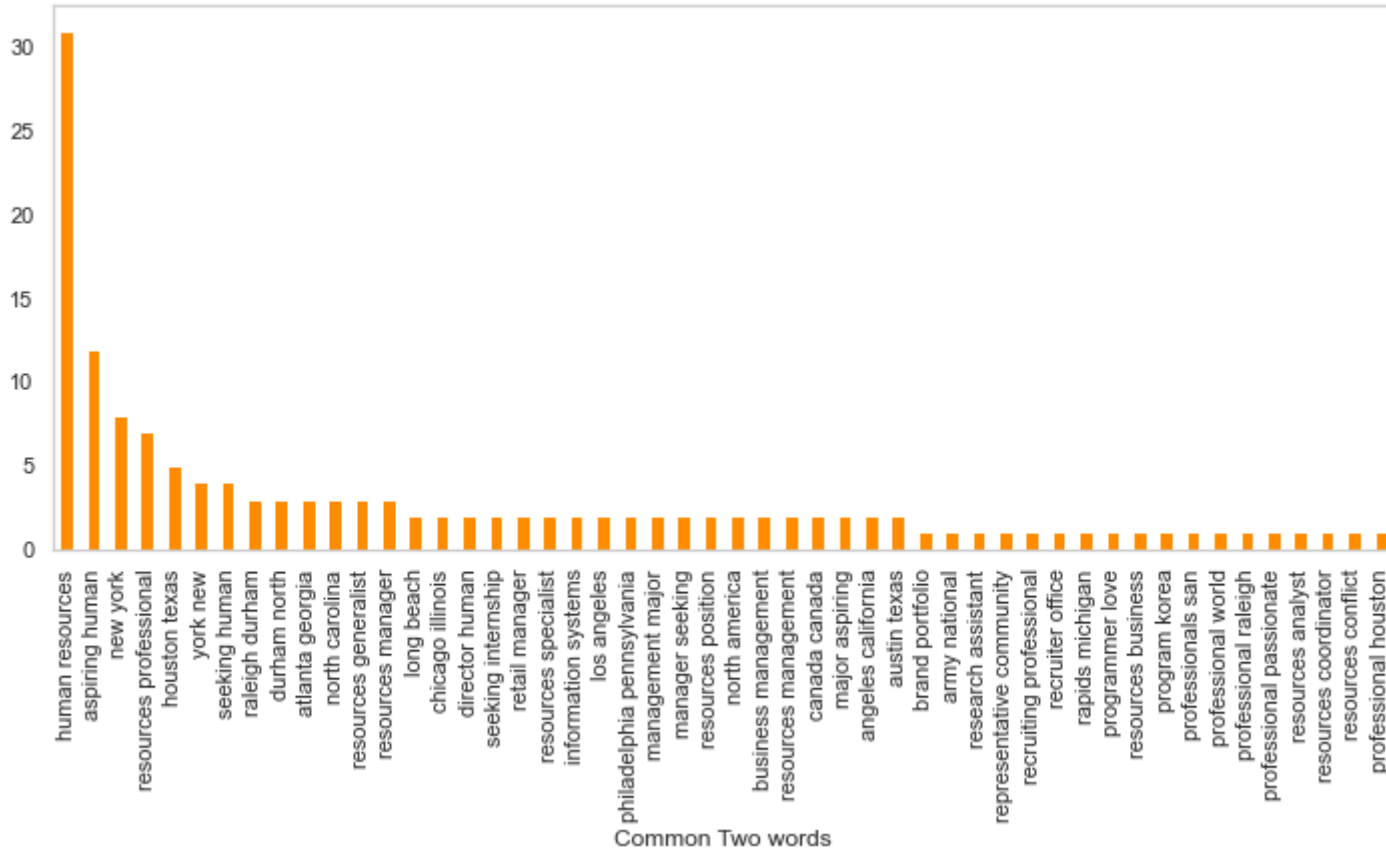
'human' and 'resources' are the most frequently occurring word. Only top 50 words are shown in the diagram.



3.1.3 Bi-gram Term frequencies

In order to better understand words that can be pairs of two consecutive words, plotting bigrams here can be useful to understand whether we need to consider consecutive words like 'human resources' or 'new york' together as a term in our modelling to get better results

Bi-gram - Common Two-word Frequency



4. Data Modeling

Data modeling is done using Gensim library, which is an open source library in python written by Radim Rehurek and is used in unsupervised topic modelling and natural language processing. </br>

It is designed to extract semantic topics from documents. It can handle large text collections. Hence it makes it different from other machine learning software packages which target memory processing. Gensim also provides efficient multicore implementations for various algorithms to increase processing speed. </br>

Gensim allows to build corpus (collection of texts) and dictionaries using simple classes and functions.

A Sample of prepared text after preprocessing

```
['2019 ct bauer college business graduate magna cum laude aspiring hum  
an resources professional houston texas'  
'native english teacher epik english program korea canada canada'  
'aspiring human resources professional raleigh durham north carolina'  
'people development coordinator ryan denton texas'  
'aspiring human resources specialist new york new york'  
'student humber college aspiring human resources generalist canada ca  
nada'  
'hr senior specialist san francisco california'  
'seeking human resources hris generalist positions philadelphia penns  
ylvania'  
'student chapman university lake forest california'  
'svp chro marketing communications csr officer engie houston woodland  
s energy gphr sphr houston texas']
```

4.1 Generate Document Vectors

Representing documents numerically gives us the ability to perform meaningful analytics and also creates the instances on which machine learning algorithms operate.

Following are the steps needed to vectorize the text

4.1.2 Tokenize words

Splits texts into tokens or words

```
[['2019', 'ct', 'bauer', 'college', 'business', 'graduate', 'magna',  
'cum', 'laude', 'aspiring', 'human', 'resources', 'professional', 'hou  
ston', 'texas'], ['native', 'english', 'teacher', 'epik', 'english',  
'program', 'korea', 'canada', 'canada'], ['aspiring', 'human', 'resour  
ces', 'professional', 'raleigh', 'durham', 'north', 'carolina'], ['peo  
ple', 'development', 'coordinator', 'ryan', 'denton', 'texas'], ['aspi  
ring', 'human', 'resources', 'specialist', 'new', 'york', 'new', 'yor  
k'], ['student', 'humber', 'college', 'aspiring', 'human', 'resource  
s', 'generalist', 'canada', 'canada'], ['hr', 'senior', 'specialist',  
'san', 'francisco', 'california'], ['seeking', 'human', 'resources',  
'hris', 'generalist', 'positions', 'philadelphia', 'pennsylvania'],  
'student', 'chapman', 'university', 'lake', 'forest', 'california'],  
'svp', 'chro', 'marketing', 'communications', 'csr', 'officer', 'engi  
e', 'houston', 'woodlands', 'energy', 'gphr', 'sphr', 'houston', 'texa  
s']]
```

4.1.3 Create bigram

Topic models make more sense when 'New' and 'York' are treated as 'New York' - we can do this by creating a bigram model and modifying our corpus accordingly.

Scoring for bigram phrases, the default is the PMI-like scoring as described in Mikolov, et. al: "Distributed Representations of Words and Phrases and their Compositionality".

```
aspiring_human 15.31720430107527
resources_professional 12.532258064516128
houston_texas 34.53333333333333
canada_canada 32.375
raleigh_durham 115.11111111111111
north_carolina 51.800000000000004
resources_specialist 4.17741935483871
new_york 56.65625
resources_generalist 8.35483870967742
seeking_human 5.56989247311828
philadelphia_pennsylvania 129.5
human_resources 16.170655567117585
atlanta_georgia 86.33333333333333
resources_management 2.78494623655914
seeking_internship 28.777777777777775
chicago_illinois 86.33333333333333
retail_manager 37.0
austin_texas 25.900000000000002
director_human 5.56989247311828
```

```
north_america 51.800000000000004
manager_seeking 8.222222222222221
business_management 17.266666666666666
major_aspiring 14.388888888888888
resources_manager 4.774193548387097
information_systems 129.5
los_angeles 129.5
management_major 28.777777777777775
long_beach 64.75
resources_position 5.56989247311828
```

4.1.4 Create Dictionary

Creates a dictionary from the text containing the number of times a word appears in the training set. This creates a mapping for Ids for each token

```
Dictionary(234 unique tokens: ['2019', 'aspiring_human', 'bauer', 'business', 'college']...)
```

4.1.5 Create a Bag of Words Corpus

Bag of words corpus in the Gensim library are based on dictionaries and contain the ID of each word along with the frequency of occurrence of the word.

```
[['2019', 1),  
 ('aspiring_human', 1),  
 ('bauer', 1),  
 ('business', 1),  
 ('college', 1),  
 ('ct', 1),  
 ('cum', 1),  
 ('graduate', 1),  
 ('houston_texas', 1),  
 ('laude', 1),  
 ('magna', 1),  
 ('resources_professional', 1)],  
 [('canada_canada', 1),  
 ('english', 2),  
 ('epik', 1),  
 ('korea', 1),  
 ('native', 1),  
 ('program', 1),  
 ('teacher', 1)],  
 [('aspiring_human', 1),  
 ('resources_professional', 1),  
 ('north_carolina', 1),  
 ('raleigh_durham', 1)]]
```

Number of unique tokens: 234

4.1.5 Create a TF-IDF matrix with Gensim

The tf-idf model transforms vectors from the bag-of-words representation to a vector space where the frequency counts are weighted according to the relative rarity of each word in the corpus.

During this transformation, it will take a vector and return another vector of the same dimensionality, except that features which were rare in the training corpus will have their value increased.

4.2 Topic Modeling

Topic Modeling refers to the probabilistic modeling of text documents as topics. Topic modeling is a family of techniques that can be used to describe and summarize the documents in a corpus according to a set of latent "topics".

4.2.1 Latent Semantic Indexing (LSI)

This transforms documents from either bag-of-words or (preferably) TfIdf-weighted space into a latent space of a lower dimensionality.

LSA or LSI learns latent topics by performing a matrix decomposition on the BoW using Singular value decomposition (SVD).

Initially the model is assigned 10 topics

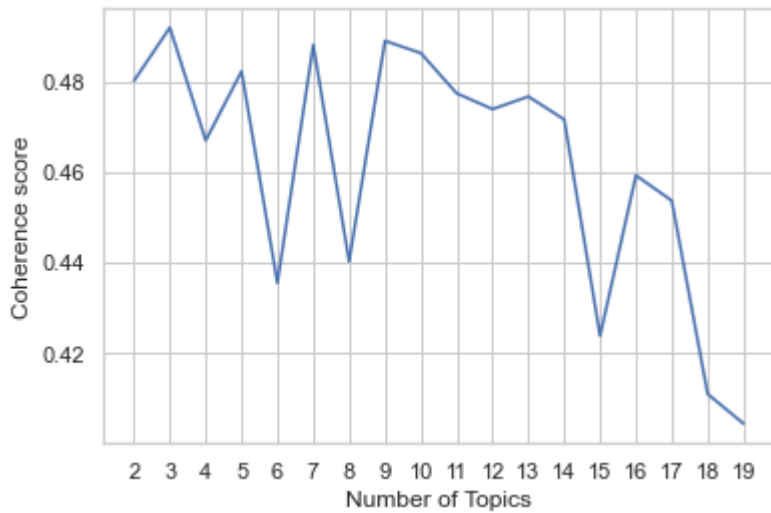
```
[(0,
  '-0.470*"new_york" + -0.357*"aspiring_human" + -0.313*"houston_texas" + -0.249*"student" + -0.238*"north_carolina" + -0.238*"raleigh_durham" + -0.234*"resources_professional" + -0.226*"seeking_internship" + -0.169*"human_resources" + -0.134*"resources_manager"'),
 (1,
  '-0.682*"new_york" + 0.330*"houston_texas" + 0.295*"student" + 0.214*"seeking_internship" + -0.173*"resources_specialist" + -0.168*"luxottica" + -0.145*"specialist" + 0.136*"resources_management" + 0.127*"north_carolina" + 0.127*"raleigh_durham"'),
 (2,
  '0.446*"raleigh_durham" + 0.446*"north_carolina" + -0.327*"houston_texas" + 0.325*"generalist" + -0.305*"student" + 0.213*"loparex" + -0.200*"seeking_internship" + 0.173*"inc" + 0.173*"scottmadden" + 0.152*"resources_professional"'),
 (3,
  '-0.400*"resources" + -0.333*"seeking_human" + -0.238*"california" + -0.232*"chicago_illinois" + -0.221*"opportunities" + -0.185*"director_human" + -0.162*"san" + -0.153*"atlanta_georgia" + -0.149*"ey" + -0.13
```

```
9*"hr"'),  
(4,  
  '-0.274*"resources" + 0.257*"california" + 0.247*"professional" + 0.  
233*"human_resources" + -0.227*"seeking_human" + 0.191*"manager" + 0.1  
78*"hr" + 0.174*"massachusetts" + -0.169*"chicago_illinois" + 0.160*"l  
os_angeles"')]
```

4.2.1.1 Determining optimum number of topics

Topic coherence measure is a realistic measure for identifying the number of topics. It uses the latent variable models. Each generated topic has a list of words. In topic coherence measure, you will find average/median of pairwise word similarity scores of the words in a topic. The high value of topic coherence score model will be considered as a good topic model.

Topic Coherence measures score a single topic by measuring the degree of semantic similarity between high scoring words in the topic. These measurements help distinguish between topics that are semantically interpretable topics and topics that are artifacts of statistical inference.



The optimum number of topics is 8 as shown in the graph

4.2.2 Latent Dirichlet Allocation (LDA)

Another transformation from bag-of-words counts into a topic space of lower dimensionality, LDA is a probabilistic extension of LSA (also called multinomial PCA), so LDA's topics can be interpreted as probability distributions over words.

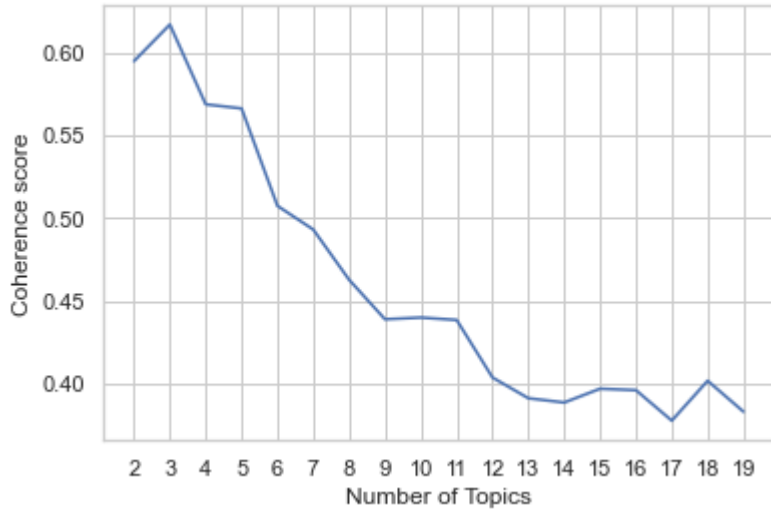
LDA uses dirichlet priors for the document-topic and word-topic distributions, lending itself to better generalization.

```
[ (0,  
  '0.024*"professional" + 0.020*"management" + 0.018*"mississippi" +
```

```
0.018*"recruiting" + 0.018*"jackson"),  
(1,  
  '0.031*"new_york" + 0.025*"indiana" + 0.025*"resources_specialist" +  
0.014*"delphi" + 0.014*"kokomo"'),  
(2,  
  '0.022*"north_carolina" + 0.022*"raleigh_durham" + 0.021*"new_york"  
+ 0.019*"resources_professional" + 0.018*"business"'),  
(3,  
  '0.035*"student" + 0.028*"houston_texas" + 0.020*"forest" + 0.020*"c  
hapman" + 0.020*"lake"'),  
(4,  
  '0.029*"seeking_internship" + 0.021*"houston_texas" + 0.021*"human_r  
esources" + 0.019*"new_york" + 0.018*"specialist"'),  
(5,  
  '0.021*"management_major" + 0.021*"milpitas" + 0.020*"english" + 0.0  
19*"experienced" + 0.016*"retail_manager"'),  
(6,  
  '0.022*"atlanta_georgia" + 0.019*"ey" + 0.016*"director_human" + 0.0  
16*"humber" + 0.016*"resources_generalist"'),  
(7,  
  '0.018*"always" + 0.018*"set" + 0.018*"success" + 0.016*"official" +  
0.016*"illinois"'),  
(8,  
  '0.021*"hris" + 0.021*"positions" + 0.018*"philadelphia_pennsylvani  
a" + 0.016*"generalist" + 0.015*"patient"'),  
(9,
```


```
'0.022*"long_beach" + 0.018*"inc" + 0.018*"scottmadden" + 0.017*"tea  
mfocused" + 0.017*"energetic"')]
```

4.2.1.1 Determining optimum number of topics



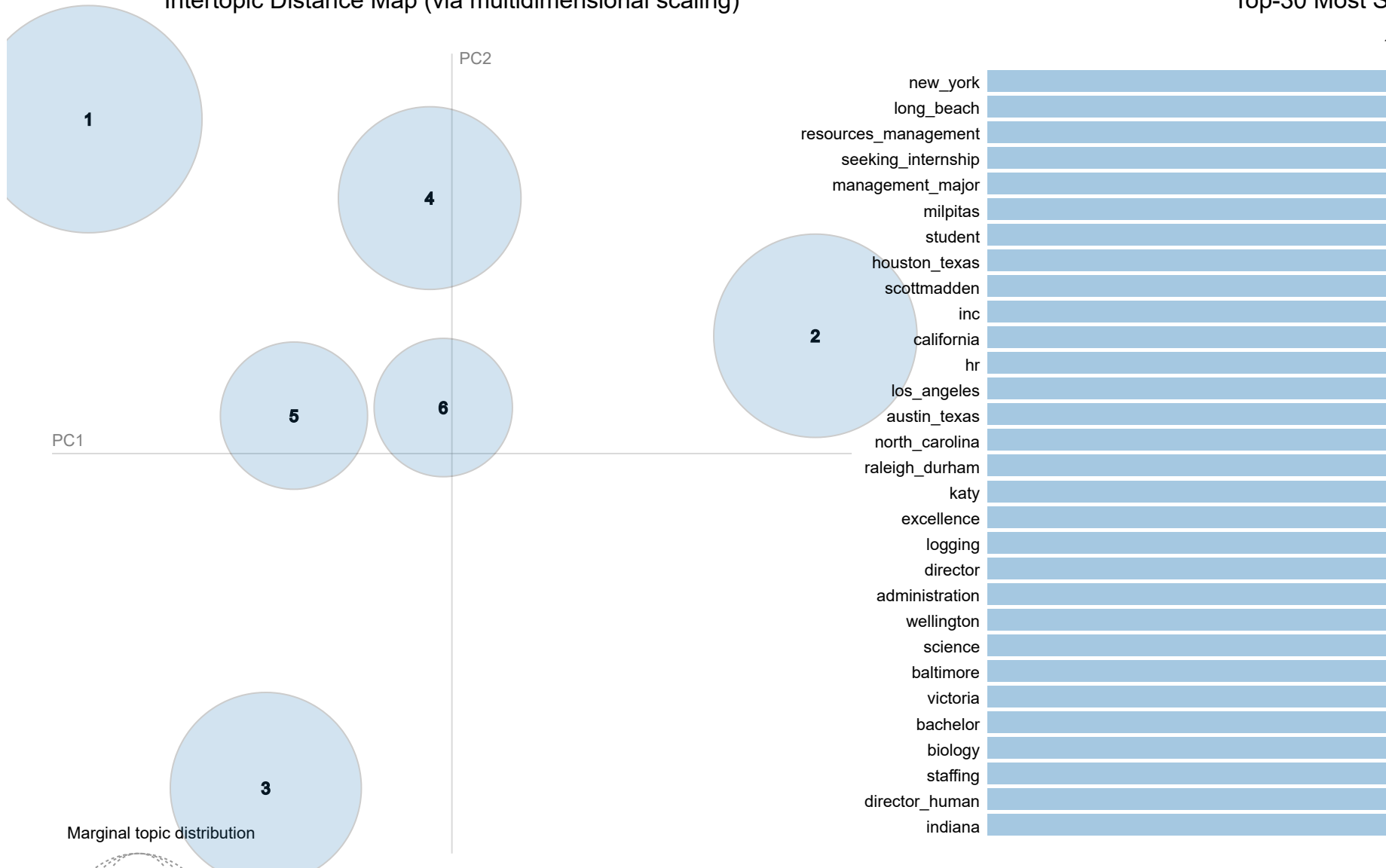
4.2.1.2 Visualization with PyLDAvis

Selected Topic:

Slide to adjust relevance metric:⁽²⁾ 
 $\lambda = 1$ 0.0

Intertopic Distance Map (via multidimensional scaling)

Top-30 Most Salient



Interpret pyLDAvis's output

- A good topic model will have fairly big, non-overlapping bubbles scattered throughout the chart instead of being clustered in one quadrant.</br>
- Each bubble on the left-hand side plot represents a topic. The larger the bubble, the more prevalent is that topic.</br>
- Blue bars represent the overall frequency of each word in the corpus. If no topic is selected, the blue bars of the most frequently used words will be displayed.</br>
- Red bars give the estimated number of times a given term was generated by a given topic
- The word with the longest red bar is the word that is used the most by the texts belonging to that topic.

4.3 Testing model on unseen data with LDA

Cosine similarity is used to determine the similarity of two vectors. Cosine similarity is a standard measure in Vector Space Modeling

		combined	doc_topic
0	2019 ct bauer college business graduate magna cum laude aspiring human resources professional houston texas		4
1	native english teacher epik english program korea canada canada		5
2	aspiring human resources professional raleigh durham north carolina		1
3	people development coordinator ryan denton texas		2
4	aspiring human resources specialist new york new york		1


```
[(0, 0.01952265), (1, 0.025141822), (2, 0.03422916), (3, 0.03139716),  
(4, 0.017879765), (5, 0.87182945)]
```

The unseen text has a higher probability with topic 5

		combined	doc_topic
1	native english teacher epik english program korea canada canada		5
6	hr senior specialist san francisco california		5
7	seeking human resources hr is generalist positions philadelphia pennsylvania		5
10	human resources coordinator intercontinental buckhead atlanta atlanta georgia		5
20	aspiring human resources manager seeking internship human resources houston texas		5

The top documents that most similar to the unseen data are shown above

4.2 Doc2Vec

Doc2Vec is a Model that represents each Document as a Vector.

In Gensim, Paragraph Vector model is referred as Doc2Vec.

Le and Mikolov in 2014 introduced the Doc2Vec algorithm

https://cs.stanford.edu/~quocle/paragraph_vector.pdf, which usually outperforms such simple-averaging of Word2Vec vectors.

There are two implementations:

Paragraph Vector - Distributed Memory (PV-DM) Paragraph Vector - Distributed Bag of Words (PV-DBOW) PV-DM is analogous to Word2Vec CBOW. The doc-vectors are obtained by training a neural network on the synthetic task of predicting a center word based an average of both context word-vectors and the full document's doc-vector.

PV-DBOW is analogous to Word2Vec SG. The doc-vectors are obtained by training a neural network on the synthetic task of predicting a target word just from the full document's doc-vector. (It is also common to combine this with skip-gram testing, using both the doc-vector and nearby word-vectors to predict a single target word, but only one at a time.)

To train the model, need to associate a tag/number with each document of the training corpus.

4.2.1 Create Tags

```
[TaggedDocument(words=['2019', 'ct', 'bauer', 'college', 'business',  
'graduate', 'magna', 'cum', 'laude', 'aspiring', 'human', 'resources',  
'professional', 'houston', 'texas'], tags=[0]),  
 TaggedDocument(words=['native', 'english', 'teacher', 'epik', 'englis  
h', 'program', 'korea', 'canada', 'canada'], tags=[1]),  
 TaggedDocument(words=['aspiring', 'human', 'resources', 'professiona  
l', 'raleigh', 'durham', 'north', 'carolina'], tags=[2]),  
 TaggedDocument(words=['people', 'development', 'coordinator', 'ryan',  
'denton', 'texas'], tags=[3]),  
 TaggedDocument(words=['aspiring', 'human', 'resources', 'specialist',  
'new', 'york', 'new', 'york'], tags=[4]))]
```

4.2.2 Training the Model

Now, we'll instantiate a Doc2Vec model with a vector size with 50 dimensions and iterating over the training corpus 40 times. We set the minimum word count to 2 in order to discard words with very few occurrences. (Without a variety of representative examples, retaining such infrequent words can often make a model worse!) Typical iteration counts in the published Paragraph Vector paper https://cs.stanford.edu/~quocle/paragraph_vector.pdf results, using 10s-of-thousands to millions of docs, are 10-20. More iterations take more time and eventually reach a point of diminishing returns.

However, this is a very very small dataset (49 documents) with shortish documents (a few hundred words). Adding training passes can sometimes help with such small datasets.

4.2.2.1 Build a vocabulary

Essentially, the vocabulary is a list (accessible via `model.wv.index_to_key`) of all of the unique words extracted from the training corpus. Additional attributes for each word are available using the `model.wv.get_vecattr()` method

4.2.2.2 Train the model

Train the model on the corpus. If optimized Gensim (with BLAS library) is being used, this should take no more than 3 seconds. If the BLAS library is not being used, this should take no more than 2 minutes, so use optimized Gensim with BLAS

4.2.2.3 Assessing the model

To assess our new model, we'll first infer new vectors for each document of the training corpus, compare the inferred vectors with the training corpus, and then returning the rank of the document based on self-similarity. Basically, we're pretending as if the training corpus is some new unseen data and then seeing how they compare with the trained model. The expectation is that we've likely overfit our model (i.e., all of the ranks will be less than 2) and so we should be able to find similar documents very easily. Additionally, we'll keep track of the second ranks for a comparison of less similar documents.

```
Counter({0: 32, 1: 6, 2: 4, 3: 2, 5: 1, 7: 1, 6: 1, 11: 1, 12: 1})
```

4.2.2.4 Getting other similar documents

```
Test Data : TaggedDocument(['aspiring', 'human', 'resources', 'profes  
sional', 'raleigh', 'durham', 'north', 'carolina'], [2])  
0.9879245162010193 | hr manager endemol shine north america los angele  
s california  
0.9846591949462891 | seeking human resources opportunities chicago ill  
inois  
0.9822432994842529 | seeking employment opportunities within customer  
service patient care torrance california  
0.9812532663345337 | bachelor science biology victoria university well  
ington baltimore maryland  
0.9765986800193787 | always set success los angeles california  
0.9731391072273254 | senior human resources business partner heil envi  
ronmental chattanooga tennessee  
0.9726248979568481 | people development coordinator ryan denton texas
```