# Text analysis, Clustering, and Topic modeling

# IDS 572 Assignment 6

Group Details

| | |
|---|---|
| **NagaShrikanth Ammanabrolu** | **676837954** |
| **Suresh Sappa** | **667192596** |
| **Sagar Kanchi** | **669850639** |

## Table of Contents

**Question No. 1)**
Solution.

**Building a Document-Term Matrix**

We begin the analysis by building a Document-Term Matrix out of the entire dataset of job postings related to Analytics.

To begin with building a Document-Term Matrix, the dataset is first imported into Rapidminer and the appropriate labels are assigned to the variables. 'File_ID' is assigned the ID attribute, Label is assigned the 'Label' attribute and the Description is defined to be 'Text'. Once the dataset is imported, we can build a document-term matrix either on the entire dataset or a subset of it by applying the 'Sampling' operator and specifying a ratio of dataset to be considered for building the document-term matrix.

Since we'll be performing clustering on the dataset later, including the entire dataset would be computationally intensive. Therefore, while performing the Clustering, we would only take a Sample of the dataset. However, for the purpose of cleaning and exploring the dataset, we've decided to build a document-term matrix on the entire dataset without using the 'Sampling' operator. The dataset is now connected to the 'Process documents from Data' operator since our dataset is stored inside a single data file and not in labeled folders. Inside this operator, we have to perform certain text mining analysis to make sure that our document-term matrix is meaningful. Below are the set of operations we perform before obtaining a document-term matrix:

1. **Tokenize –** We split each of the document (a job post) into tokens (sets of words)
2. **Eliminate Stopwords –** More generally occurring terms are eliminated
3. **Lowercase –** To avoid duplicate tokens, we lowercase the entire dataset
4. **Filter the tokens by their length –** We filter the tokens where they have character length between 4 and 25
5. **Pruning the Dataset –** We also prune the dataset where we ignore words that appear in less than 2% of the entire dataset or appear in more than 90% of the entire dataset. This specific step is to make sure we eliminate generalized and specific terms related to a subset of job postings

Once we setup this process, we obtain a document-term matrix with all of the documents and their TF-IDF measure across all of the terms in the dataset post-processing. If a document-term matrix on a sample of the dataset was required, we could've made use of the Sampling operator and obtain a document-term matrix on the sampled dataset.

**Eliminating Terms unusable for our Analysis**

The entire process that we've carried above eliminated all of the stopwords and other terms based on their character lengths. After performing the above steps, we are still left with a few keywords which are either typos or don't help us in our analysis. For this step, we would have to take a look at the document-term matrix and look for words which we believe are unusable for our Analysis. This process requires that we have a document-term matrix on the entire dataset to better understand and eliminate redundant words in our dataset.

We can perform this step in Rapidminer by first examining the document-term matrix for such words. We create a list of these unusable words into a new Text document. The 'Filter Tokens (By Content)' operator in Rapidminer allows users to specify a list of Custom Stopwords as Strings using the concept of Regular expressions. We made sure to select the 'Invert Selection' option, so that the list of words inputted to this operator are eliminated from the analysis. This list of custom stopwords has to be updated manually and some human context is required in this step to make meaningful decisions.

As we perform the Topic modeling analysis in MALLET later, we also find that it too does take in a list of custom stopwords through the method '--remove-stopwords'. In addition to getting rid of the Stopwords or based on the character length of tokens, a custom stopwords is required in most of the text analysis to eliminate additional terms which are unusable for further analysis.

**Data Exploration: Summarizing the Data**

Out of the entire 23,223 job postings related to Analytics in the dataset, below is a distribution of the different categories of job postings and their fraction in the entire dataset.

| Distribution of Job Posting Labels | | |
|---|---|---|
| **Label** | **Number of Job postings** | **Overall Fraction of Jobs** |
| **Analytics** | 3899 Jobs | 16.79% |
| **Data Mining** | 3780 Jobs | 16.28% |
| **Big Data** | 3258 Jobs | 14.03% |
| **Business Analytics** | 2834 Jobs | 12.20% |
| **Data Analytics** | 2217 Jobs | 9.54% |
| **Business Intelligence** | 2090 Jobs | 9.00% |
| **Data Science** | 1810 Jobs | 7.79% |
| **Machine Learning** | 1803 Jobs | 7.76% |
| **Predictive Modeling** | 1526 Jobs | 6.57% |

From the above table, we can see that job postings with a label of 'Analytics', 'Data Mining' and 'Big Data' make up for 47% of the entire job postings in our dataset.

The size of the document-term matrix is: (23217, 1426)

Hence, we obtain the said Descriptive statistics on the given dataset. The next steps are to perform (a) Clustering of the Job postings and (b) Topic modeling of the job postings to better understand the content of these job postings and other factors associated with these job listings.

*(The Distribution of Jobs by their associated labels is shown in the Appendix)*

**Question No. 2)**
**Solution.**

**Performing Clustering and Interpreting the Clusters**

The next step in our analysis is to obtain a relevant set of clusters which will try to accurately describe the clusters of job postings and how they are related to each other. For this purpose, we will be using Sampling here, since taking the entire dataset into clustering will be computationally intensive and time-consuming. For this purpose, we perform clustering on a subset of the dataset, determine the 'best' value of the clustering model and then perform it on the entire dataset. We take a sample of the 10% of the dataset for performing clustering analysis. This gives us a total of 2,321 items on which we perform our clustering analysis.

For our first clustering model, we specify the number of clusters as 20. Below is a summary of the interpretation of the resulting 20 clusters through this model.

| Interpretation of K=20 Clustering model | | |
|---|---|---|
| **Cluster Number** | **Number of Items** | **Keyword across Documents** |
| Cluster 0 | 114 items | Sales |
| Cluster 1 | 76 items | Algorithms/Databases |
| Cluster 2 | 84 items | Project Management |
| Cluster 3 | 34 items | Process/Systems manager |
| Cluster 4 | 84 items | Risk Analytics |
| Cluster 5 | 228 items | Supporting existing systems |
| Cluster 6 | 86 items | Data Architecture |
| Cluster 7 | 133 items | Data Science/Machine Learning |
| Cluster 8 | 119 items | Healthcare |
| Cluster 9 | 214 items | Systems Architect |
| Cluster 10 | 148 items | Software Developer |
| Cluster 11 | 188 items | Marketing/Communications |
| Cluster 12 | 46 items | Operations Management |
| Cluster 13 | 249 items | Application architect |

| | | |
|---|---|---|
| Cluster 14 | 82 items | Financial analyst |
| Cluster 15 | 171 items | Semantic Analytics |
| Cluster 16 | 28 items | Search Analytics |
| Cluster 17 | 45 items | Research Analyst |
| Cluster 18 | 155 items | Predictive Modeling |
| Cluster 19 | 37 items | Mobile Analytics |

Now, this is one way of interpreting the clusters. But since the Number of topics is a user-specified term and we will be working with k as 10, 20, 30, 40 and 50, this is not a better way of interpreting the clustering outcome.

For interpreting the clusters, we will take a look at the top 10 words which represent each of the cluster and if they are meaningful, the cluster sizes, within cluster distances and the between cluster distances. After performing the clustering, we will take a look at the top 15-20 words representing each cluster and eliminate words which we do not find are useful for our analysis.

Below is an interpretation of the clusters generated by a 'k' value of 20.

| **Performance Vector of Clusters** | | **Cluster Model (% share)** | |
|---|---|---|---|
| **Average within Centroid Distance** | 0.847 | **Cluster 0** | 5.04% |
| **Average within Centroid Distance (Cluster 0)** | 0.777 | **Cluster 1** | 4.95% |
| **Average within Centroid Distance (Cluster 1)** | 0.825 | **Cluster 2** | 1.68% |
| **Average within Centroid Distance (Cluster 2)** | 0.866 | **Cluster 3** | 3.74% |
| **Average within Centroid Distance (Cluster 3)** | 0.837 | **Cluster 4** | 3.61% |
| **Average within Centroid Distance (Cluster 4)** | 0.798 | **Cluster 5** | 9.34% |
| **Average within Centroid Distance (Cluster 5)** | 0.886 | **Cluster 6** | 3.66% |
| **Average within Centroid Distance (Cluster 6)** | 0.800 | **Cluster 7** | 3.79% |
| **Average within Centroid Distance (Cluster 7)** | 0.860 | **Cluster 8** | 2.88% |
| **Average within Centroid Distance (Cluster 8)** | 0.834 | **Cluster 9** | 7.32% |
| **Average within Centroid Distance (Cluster 9)** | 0.859 | **Cluster 10** | 4.30% |
| **Average within Centroid Distance (Cluster 10)** | 0.858 | **Cluster 11** | 5.73% |
| **Average within Centroid Distance (Cluster 11)** | 0.782 | **Cluster 12** | 4.00% |
| **Average within Centroid Distance (Cluster 12)** | 0.842 | **Cluster 13** | 11.07% |
| **Average within Centroid Distance (Cluster 13)** | 0.879 | **Cluster 14** | 4.56% |
| **Average within Centroid Distance (Cluster 14)** | 0.844 | **Cluster 15** | 6.07% |
| **Average within Centroid Distance (Cluster 15)** | 0.886 | **Cluster 16** | 4.82% |
| **Average within Centroid Distance (Cluster 16)** | 0.816 | **Cluster 17** | 4.35% |
| **Average within Centroid Distance (Cluster 17)** | 0.868 | **Cluster 18** | 6.54% |
| **Average within Centroid Distance (Cluster 18)** | 0.853 | **Cluster 19** | 2.45% |
| **Average within Centroid Distance (Cluster 19)** | 0.849 | | |
| **Davies Bouldin Index** | 5.864 | | |

From the above table, we can conclude that a cluster size of 20 does yield us good clusters based on the within cluster distance and overall cluster sizes.

**Evaluative comparison of the different Best models**

Now that we have obtained an interpretation of the clusters from our clustering model as described above, we now need to consider other terms for the 'Number of clusters' variable. We will consider k as 10, 20, 30, 40 and 50, to compare across all of the output clusters and choose our best model accordingly.

*(The process of obtaining different clusters and interpreting them is Summarized in the Appendix.)*

In-addition to the numerical analysis of within cluster distance and the cluster sizes, we've also taken into account the between cluster distances. Through an evaluative comparison of the different obtained clusters, we can conclude that clustering with the value of 'k' as 50 or 50 Topics yields us our best model based on the k-Means clustering algorithm. A model with the value of 'k' as 50 gives us best possible clusters with low within cluster distance and high inter-cluster distances. These values are also close, signaling towards compact clusters.

Once we've obtained our best cluster, we performed the same on the entire dataset and obtained 50 clusters on the entire dataset. The sheer volume of the clusters did not allow us to manually check the job listings in each cluster, although we checked a few of the listings in each cluster and if they matched. From this process, we concluded that performing Clustering does yield us meaningful results in our given dataset.

**Measures used for Word-Vector and Document Distances**

**Measure used for Document-Term Matrix**: We have used the Term Frequency–Inverse Document Frequency measure for the creation of our Document-Term matrix. The measure of TF-IDF is given as a product of a term's term frequency and its Inverse-document frequency. TF-IDF gives us a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. [1]

The goal with TF-IDF is to model each document into a vector space, ignoring the exact ordering of the words in the document while retaining information about the occurrences of each word. The TF-IDF gives us how important is a word to a document in a collection, since it takes in consideration not only the isolated term but also the term within the document collection. For instance, a term that occurs 10 times more than another isn't 10 times more important than it. [2]

Also, since we are dealing with vectors, which only take in numbers as input and not text, we need a measure to convert the text into its appropriate numerical measure using TF-IDF.

**Measure used for Document Distances:** Since the measure of Euclidean distance is very large for document-term vectors of varied lengths, we can't use it for measuring document distances. Even for documents with an almost similar distribution, the Euclidean measure yields us a high value. Another aspect that could be considered to measure distance between documents is the angle between them. For two similar documents, we will yield a high value through Euclidean distance, but a value of 0 if we measure the angle between them which is accurate.

In Cosine Similarity measure, each term is assigned a different dimension and a document is characterized by a vector where the value of each dimension corresponds to the number of times that term appears in the document. Cosine similarity is very efficient to evaluate, especially for sparse vectors, as only non-zero dimensions need to be considered. [3]

Hence, we will use the numerical measure of Cosine Similarity to measure distances between documents.

**Question No. 3) (a)**
**Solution.**

**What is Topic Modeling?**

Given a large corpus of 'Unstructured data', to perform text mining on such data, Topic modeling is one of the most frequently used concept. Topic modeling basically deals with the clustering of frequently occurring and related words together. Topic modeling deals with the clustering of words in a topic based on their probabilities of co-occurrence. Such a formulation of topics in any given document allows users to find words that occur in a similar context together and how they are distinguished from other words/terms in the document.

Topic models are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. [4] Inferring a document or even the creation of a new document based on this information becomes easy with topic modeling. The probabilistic distribution of the terms over a particular topic makes it easy to interpret a topic more precisely as compared to the spatial representation through regular clustering techniques.

In Topic modeling, the LDA (Latent Dirichlet Allocation) model compares the occurrence of topics within a document to how a word has been assigned in other documents to find the best match. [5] Topic modeling deals with the inferring of particular topics that might have aided in the creation of a document based on the occurrence of the words in a particular passage of the document. For a particular set of words which have a possibility of co-occurring across different sets of topics, the probability factor of the occurrence of a word in a topic would come handy here. Say, for example, the word "lead" could across two topics where one means a leadership quality and in the other topic means a poisoning metal. Based on the document, the probability measure of this term across a topic would help us infer the context in which this term occurs in the topic.

In the implementation of Topic modeling using LDA, the set of words are assigned to topics randomly and then we just keep improving the model, to make sure the guesses are more internally consistent, until the model reaches an equilibrium that is as consistent as the collection allows. This equilibrium across topic modeling is achieved through a measurement called 'log-likelihood' function. Running a topic modeling algorithm with a specified number of iterations aims at reducing this value of the 'log-likelihood' estimate to obtain an equilibrium-state, where the terms within topics are somehow more related and more distinguished across different topics.

**Question No. 3) (b)**
**Solution.**

**Primary Objective in using Topic Modeling**

Given our large corpus of 'Unstructured data' in the form of job postings, our primary objective is to identify the patterns in keywords across these listings. Since topic modeling allows us to classify terms into different topics, we can later inspect these topics and derive insights as to what these topics signify. Although we have performed clustering models on the job listings earlier, these spatial representation of terms across documents is not enough to derive insights. A word's mere presence in a topic does not signify much and is not sufficient information to draw conclusions out of.

Making use of Topic modeling allows us to classify the terms into different topics, but with an additional factor of probability, which plays a key role here. We not only would like to know how the terms in our documents are scattered across different topics, we also want to know the probabilities associated with each of the term of being in a particular topic. All of this is made possible only through topic modeling and not a general clustering algorithm. Hence, to facilitate our needs to adequately represent the terms into different topics along with their probabilities, we have to make use of Topic modeling given our vast unstructured dataset.

**Question No. 3) (c)**
Solution.

**Defining the 'Vocabulary' used to perform Topic Modeling**

To define the 'Vocabulary' that we will be using to build our topic models, we would need to run our models iteratively and improve upon the list of terms we wish to eliminate from our analysis. To begin with, we first derive a '.mallet' file from our entire dataset through the MALLET tool. Once we have obtained this '.mallet' file, we next input this file, along with a list of custom stopwords to eliminate from our analysis. After we run the analysis, we obtain 'Topic-Keys' and 'Document-Topics matrix' pair.

To improve upon our custom list of stopwords to eliminate, we look at the 'Topic-Keys' output file. Each of the topic is given as a pair of keys/terms that define the particular topic. We would look for typos, non-understandable words and other unusable words from this file and add them to our list of custom stopwords. Also, there were situations where a particular term appeared across multiple 'Topics-Keys' generated through MALLET. We would need unique terms for each topic and words which appeared across multiple topics were updated in the 'Custom Stopwords' file.

We run the process again, iteratively, until we are left with a 'Topic-Keys' pair file which is meaningful and helps us understanding the job listings better.

**Including bi-grams in our analysis**

Making use of the '--xml-topic-report' and '--xml-topic-phrase-report' functions in MALLET, we also obtained 2-grams and 3-grams for our analysis. There's one major drawback to this process. As we've mentioned a list of custom stopwords, which are words which individually do not add value for our analysis, these words will also be eliminated from the n-grams generated in this step.

Passing in a custom list of stopwords to MALLET also causes it to eliminate it from the generated 2-grams and 3-grams. This is one major drawback to generating n-grams with MALLET. Moreover, after we obtain a list of useful combination of words like 'Software Development', which would aide our analysis further, it does require further steps in processing.

The n-grams generated in MALLET through the above-said process would need to be again added into our analysis as custom words, like, 'Software_Development' or 'softwareDevelopment'. This has caused us to terminate any further usage of n-grams for our analysis of the job listings.

**Comparing different values of Number of Topics**

We have experimented with the values for the Number of Topics in Mallet as 20, 30, 40 and 50 topics. For the purpose of evaluating the estimation of a 'better' fit, the Log-Likelihood estimate has to be checked for. It is important that our model reduces the Log-Likelihood estimate to obtain a better 'fit' for the topics. After performing the analysis on the said number of topics, we find that 20 topics yields us the lowest possible Log-Likelihood estimate as compared to that obtained under other number of topics.

While performing this analysis iteratively, as mentioned earlier, we have eliminated more words which are unusable for us through the 'Custom Stopwords' file. Also, we tried looking at some of the keywords for each topic under the said models and found that with the number of topics as 20, we find many of the topics to be useful for our further analysis.

Since we are primarily concerned with the keywords that define our topics, the number of topics as 20 yields as topics which could be easily distinguished from the others.

**Question No. 3) (d)**
**Solution.**

**Describing the obtained Topics**

From the previous step, we've obtained a list of 20 Topics and the keywords that describe each of these topics. All of these topics and their constituting keywords are not entirely useful for our analysis. Hence, we identify a set of topics which are relevant and based on their keywords, provide us further insights into what these topics might be about.

**Topic no. 12** – This topic includes keywords such as Healthcare, Health, Clinic, Drug, Research, Medical, Government, etc. in addition to analytics and data. By a quick-look at the keywords, it is safe to assume that this topic might be about Analytics positions in Healthcare.

**Topic no. 10** – This topic includes keywords such as Hadoop, Java, Python, Scaled, Distributed, Infrastructure, Cloud, etc. We can conclude that this topic is about Distributed computational applications in the field of Big Data Analytics.

**Topic no. 14** – This topic includes keywords such as Global, Management, Information, Delivery, Retail, Modeling, SAP, etc. We can conclude that this topic is about Analytics in the field of Management.

**Topic no. 5** – This topic includes keywords such as Financial, Finance, Credit, Risk, Capital, Investment, Banking, Audit, Investment, etc. We can conclude that this topic is about Analytics in the field of Finance and Trading.

**Topic no. 1** – This topic includes keywords such as Machine, Learning, Research, Language, Processing, Scientists, Programming, Algorithms, etc. We can conclude that this topic is about Analytics in the field of Machine Learning and Research.

**Topic no. 18** – This topic includes keywords such as Database, Architecture, Warehouse, Oracle, Server, Integration, etc. We can conclude that this topic is about Analytics in the field of Data Warehousing.

These are summarizing just a few of the topics that we find useful from our analysis. Some other keywords in topics are indicative of positions at a Startup, in a Web analytics role, Social media analytics role, Systems support engineer role, Marketing/Sales Analytics role, or as a Project/Process management role. Out of the total 20 topics, we've found 12 topics that would be useful for understanding the job listings.

## Checking the Validity of the Obtained Topics

For the purpose of checking the validity of the obtained topics, we will be selecting 5 of the obtained topics and take a look at a few of the job descriptions based on the Doc-Topics matrix. For this purpose, we will be considering Topic no. 1, Topic no. 5, Topic no. 10, Topic no. 12, and Topic no. 14.

**Analysis of Topic no. 10**: By sorting the job id's in the Topic 10 based on their probabilities on the Document-Topic matrix, we find the top 5 job postings for this topic. Our previous assumption that this topic includes Big Data Engineering job listings has now been confirmed. This topic involves Big Data Engineer job listings from companies like Verizon, Apple and Teradata.

**Analysis of Topic no. 1**: By sorting the job id's in the Topic 4 based on their probabilities on the Document-Topic matrix, we find the top 5 job postings for this topic. Our previous assumption that this topic includes Natural Language Processing and Machine Learning job listings has now been confirmed. This topic involves NLP job listings from Apple.

**Analysis of Topic no. 14**: By sorting the job id's in the Topic 6 based on their probabilities on the Document-Topic matrix, we find the top 5 job postings for this topic. Our previous assumption that this topic includes Analytics in the Management job listings has now been confirmed. This topic involves Scheduling and Operations Analyst job listings from companies like FrieslandCampina and more.

**Analysis of Topic no. 5**: By sorting the job id's in the Topic 12 based on their probabilities on the Document-Topic matrix, we find the top 5 job postings for this topic. Our previous assumption that this topic includes Analytics in the Finance Sector job listings has now been confirmed. This topic involves Finance and Credit Risk Analyst job listings from companies like JPMorgan and more.

**Analysis of Topic no. 12**: By sorting the job id's in the Topic 13 based on their probabilities on the Document-Topic matrix, we find the top 5 job postings for this topic. Our previous assumption that this topic includes Analytics in the Healthcare and Government sector job listings has now been confirmed. This topic involves Scientific Research Analyst job listings from organizations like Sandia, Bloomberg Government and more.

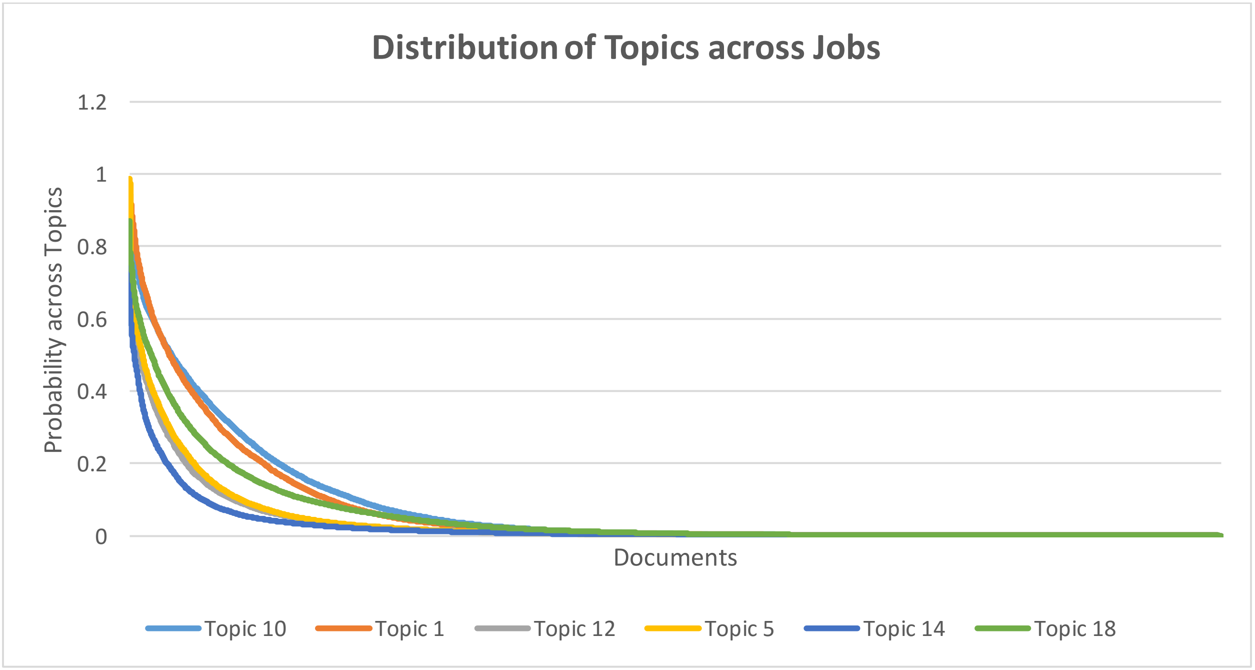
Although the Topics seem to give us accurate results and validate our previous assumptions, some of the top 5 job listings in each topic are from a same organization. When sorted by probability, many of the top listings in each topic origin from a same organization with similar job listings.

Hence, we can conclude that the topic assignment obtained does help us in understanding job listings in our given dataset better.

## Analyzing the Distribution of Topics across Jobs

To analyze the distribution of Topics across Jobs, we take a look at the 'DocTopicMatrix' file. This file allows us to keep track of how Documents are distributed across different topics, along with their probabilities. However, all of the topics do not yield us meaningful and only a subset of the generated Topics should be used to obtain this distribution.

As mentioned previously, we found the top 6 Topics which yield us best possible results, which are distinguishable from the rest. Hence, we use these 6 topics to plot the distribution of topics across jobs. This plot is given below.



From the above plot, we can see that Topics 1 and 10 are our best topics, since they yield us a distribution curve that is spread out further as compared to the other topics. All of these topics possess a high probability in the Document-Term matrix, hence adequately satisfying our conditions to obtain effective groups of Topics for describing the given Jobs.

From the Document-Term matrix, we can see that Jobs are indeed spread across different Topics, with varied probabilities. As for the count of Jobs distributed across Topics, we find that all of the Jobs are distributed across all of the topics with varied probabilities.

## Analyzing the Distribution of Keywords across Topics

To analyze the distribution of Topics across Jobs, we take a look at the 'TopicKeys' file. This file allows us to keep track of how job keywords are distributed across different topics. Since our primary aim here is to obtain Topics with keywords which are easily distinguishable from the rest, we eliminate words that appear across multiple topics.

Since we avoid generalization of keywords across multiple topics, we obtain Topic-Keys where keywords are more specific to particular Topics. For example, Topic no. 10 includes keywords such as Java, Python, Hadoop, etc., which are unique to the said topic. Similarly, we find that Topic No. 12 is more specific to keywords like Health, Healthcare, Clinical, Medicine, etc., which are specific to this particular topic. We would still find a few keywords which are shared across multiple topics, if they aren't eliminated through the remove additional stopwords function.

**Question No. 4)**
**Solution.**

The two approaches that we have used throughout this assignment was gaining insights into what the job postings were about through the usage of Clustering and Topic Modeling. By comparing the Clustering model with the Topic Modeling, we find that both of these methods are distinguished in their approaches to identifying job listings. Although there is a fine share of similarity between Clustering and Topic Modeling, both of them are closely-related and can benefit each other.

Topic modeling can project documents into a topic space which facilitates effective document clustering. Cluster labels discovered by document clustering can be incorporated into topic models to extract local topics specific to each cluster and global topics shared by all clusters. [6] Combining both topic modeling and clustering would help us discover groups in a collection of document and also to identify local topics which are specific to few clusters.

When we compare topic modeling and clustering, we notice that their approaches to grouping documents are entirely different. We also observe that clustering does not yield different groups of documents when run multiple times. But in the case of topic modeling, we get different topics each time we run the algorithm.

So, although we obtain useful information by performing both clustering and topic modeling, topic modeling helps us much better in understanding what the job ads are about. It would actually be much more beneficial to combine both the results of clustering and topic modeling to aid us better in the identifying what the job postings are about.

## REFERENCES

[1] https://en.wikipedia.org/wiki/Tf%E2%80%93idf

[2] http://aimotion.blogspot.com/2011/12/machine-learning-with-python-meeting-tf.html

[3] https://en.wikipedia.org/wiki/Cosine_similarity

[4] Mark Steyvers (2007). Probabilistic Topic Models, *Handbook of latent semantic analysis,* 427 (7), 424-440

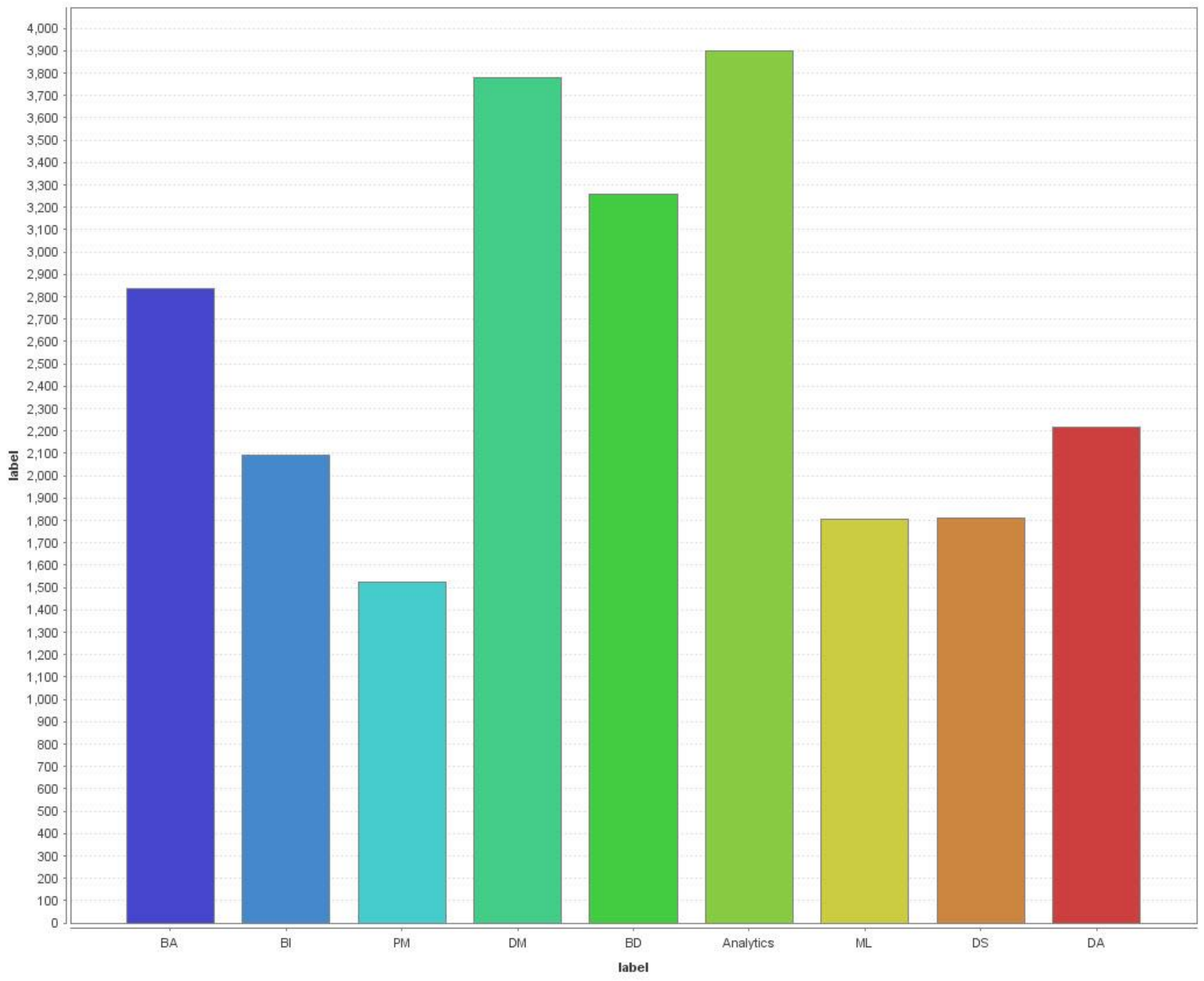
[5] Megan R. Brett (2012). Topic Modeling: A Basic Introduction, *Journal of Digital Humanities,* Vol. 2, No. 1

[6] Pengtao Xie, Eric P. Xing (2013). Integrating Document Clustering and Topic Modeling, *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence* (UAI2013), UAI-P-2013-PG-694-703

## APPENDIX

### Question No. 1

### Distribution of Jobs based on their associated Labels



Distribution of Job listings based on Labels

**Question No. 2**

**K = 10**

| Performance Vector of Clusters | | Cluster Model (% share) | |
|---|---|---|---|
| **Average within Centroid Distance** | 0.870 | **Cluster 0** | 13.09% |
| **Average within Centroid Distance (Cluster 0)** | 0.892 | **Cluster 1** | 12.66% |
| **Average within Centroid Distance (Cluster 1)** | 0.888 | **Cluster 2** | 12.58% |
| **Average within Centroid Distance (Cluster 2)** | 0.856 | **Cluster 3** | 7.32% |
| **Average within Centroid Distance (Cluster 3)** | 0.832 | **Cluster 4** | 9.43% |
| **Average within Centroid Distance (Cluster 4)** | 0.876 | **Cluster 5** | 2.06% |
| **Average within Centroid Distance (Cluster 5)** | 0.597 | **Cluster 6** | 7.15% |
| **Average within Centroid Distance (Cluster 6)** | 0.868 | **Cluster 7** | 6.41% |
| **Average within Centroid Distance (Cluster 7)** | 0.849 | **Cluster 8** | 10.72% |
| **Average within Centroid Distance (Cluster 8)** | 0.867 | **Cluster 9** | 18.52% |
| **Average within Centroid Distance (Cluster 9)** | 0.902 | | |
| **Davies Bouldin Index** | 6.509 | | |

**K = 30**

| Performance Vector of Clusters | | Cluster Model (% share) | |
|---|---|---|---|
| **Average within Centroid Distance** | 0.830 | **Cluster 0** | 3.23% |
| **Average within Centroid Distance (Cluster 0)** | 0.841 | **Cluster 1** | 5.90% |
| **Average within Centroid Distance (Cluster 1)** | 0.879 | **Cluster 2** | 3.74% |
| **Average within Centroid Distance (Cluster 2)** | 0.797 | **Cluster 3** | 1.59% |
| **Average within Centroid Distance (Cluster 3)** | 0.843 | **Cluster 4** | 3.61% |
| **Average within Centroid Distance (Cluster 4)** | 0.851 | **Cluster 5** | 1.20% |
| **Average within Centroid Distance (Cluster 5)** | 0.740 | **Cluster 6** | 1.20% |
| **Average within Centroid Distance (Cluster 6)** | 0.791 | **Cluster 7** | 3.79% |
| **Average within Centroid Distance (Cluster 7)** | 0.857 | **Cluster 8** | 2.32% |
| **Average within Centroid Distance (Cluster 8)** | 0.862 | **Cluster 9** | 2.58% |
| **Average within Centroid Distance (Cluster 9)** | 0.861 | **Cluster 10** | 6.24% |
| **Average within Centroid Distance (Cluster 10)** | 0.861 | **Cluster 11** | 5.17% |
| **Average within Centroid Distance (Cluster 11)** | 0.874 | **Cluster 12** | 4.99% |
| **Average within Centroid Distance (Cluster 12)** | 0.868 | **Cluster 13** | 2.67% |
| **Average within Centroid Distance (Cluster 13)** | 0.751 | **Cluster 14** | 2.15% |
| **Average within Centroid Distance (Cluster 14)** | 0.765 | **Cluster 15** | 2.02% |
| **Average within Centroid Distance (Cluster 15)** | 0.880 | **Cluster 16** | 1.93% |
| **Average within Centroid Distance (Cluster 16)** | 0.559 | **Cluster 17** | 1.63% |
| **Average within Centroid Distance (Cluster 17)** | 0.793 | **Cluster 18** | 4.17% |
| **Average within Centroid Distance (Cluster 18)** | 0.770 | **Cluster 19** | 4.35% |
| **Average within Centroid Distance (Cluster 19)** | 0.783 | **Cluster 20** | 1.93% |

**K = 30 (...Continued)**

| Performance Vector of Clusters | | Cluster Model (% share) | |
|---|---|---|---|
| **Average within Centroid Distance (Cluster 20)** | 0.866 | **Cluster 21** | 5.00% |
| **Average within Centroid Distance (Cluster 21)** | 0.814 | **Cluster 22** | 4.17% |
| **Average within Centroid Distance (Cluster 22)** | 0.858 | **Cluster 23** | 4.95% |
| **Average within Centroid Distance (Cluster 23)** | 0.827 | **Cluster 24** | 2.32% |
| **Average within Centroid Distance (Cluster 24)** | 0.836 | **Cluster 25** | 4.95% |
| **Average within Centroid Distance (Cluster 25)** | 0.846 | **Cluster 26** | 1.89% |
| **Average within Centroid Distance (Cluster 26)** | 0.696 | **Cluster 27** | 2.84% |
| **Average within Centroid Distance (Cluster 27)** | 0.844 | **Cluster 28** | 3.66% |
| **Average within Centroid Distance (Cluster 28)** | 0.872 | **Cluster 29** | 4.78% |
| **Average within Centroid Distance (Cluster 29)** | 0.858 | | |
| **Davies Bouldin Index** | 5.652 | | |

**K = 50**

| Performance Vector of Clusters | | Cluster Model (% share) | |
|---|---|---|---|
| **Average within Centroid Distance** | 0.812 | **Cluster 0** | 3.36% |
| **Average within Centroid Distance (Cluster 0)** | 0.750 | **Cluster 1** | 1.46% |
| **Average within Centroid Distance (Cluster 1)** | 0.762 | **Cluster 2** | 1.50% |
| **Average within Centroid Distance (Cluster 2)** | 0.825 | **Cluster 3** | 3.14% |
| **Average within Centroid Distance (Cluster 3)** | 0.838 | **Cluster 4** | 1.29% |
| **Average within Centroid Distance (Cluster 4)** | 0.815 | **Cluster 5** | 4.78% |
| **Average within Centroid Distance (Cluster 5)** | 0.872 | **Cluster 6** | 4.22% |
| **Average within Centroid Distance (Cluster 6)** | 0.849 | **Cluster 7** | 1.42% |
| **Average within Centroid Distance (Cluster 7)** | 0.834 | **Cluster 8** | 1.33% |
| **Average within Centroid Distance (Cluster 8)** | 0.843 | **Cluster 9** | 3.79% |
| **Average within Centroid Distance (Cluster 9)** | 0.836 | **Cluster 10** | 1.50% |
| **Average within Centroid Distance (Cluster 10)** | 0.846 | **Cluster 11** | 3.57% |
| **Average within Centroid Distance (Cluster 11)** | 0.856 | **Cluster 12** | 1.20% |
| **Average within Centroid Distance (Cluster 12)** | 0.782 | **Cluster 13** | 3.36% |
| **Average within Centroid Distance (Cluster 13)** | 0.783 | **Cluster 14** | 0.77% |
| **Average within Centroid Distance (Cluster 14)** | 0.821 | **Cluster 15** | 0.77% |
| **Average within Centroid Distance (Cluster 15)** | 0.754 | **Cluster 16** | 1.24% |
| **Average within Centroid Distance (Cluster 16)** | 0.855 | **Cluster 17** | 0.94% |
| **Average within Centroid Distance (Cluster 17)** | 0.753 | **Cluster 18** | 3.27% |
| **Average within Centroid Distance (Cluster 18)** | 0.780 | **Cluster 19** | 2.88% |
| **Average within Centroid Distance (Cluster 19)** | 0.777 | **Cluster 20** | 2.06% |
| **Average within Centroid Distance (Cluster 20)** | 0.824 | **Cluster 21** | 2.67% |
| **Average within Centroid Distance (Cluster 21)** | 0.826 | **Cluster 22** | 1.20% |
| **Average within Centroid Distance (Cluster 22)** | 0.840 | **Cluster 23** | 1.29% |

| | | | |
|---|---|---|---|
| **Average within Centroid Distance (Cluster 23)** | 0.692 | **Cluster 24** | |
| **Average within Centroid Distance (Cluster 24)** | 0.835 | **Cluster 25** | 1.55% |
| **Average within Centroid Distance (Cluster 25)** | 0.780 | **Cluster 26** | 2.84% |
| **Average within Centroid Distance (Cluster 26)** | 0.858 | **Cluster 27** | 1.42% |
| **Average within Centroid Distance (Cluster 27)** | 0.696 | **Cluster 28** | 4.00% |
| **Average within Centroid Distance (Cluster 28)** | 0.783 | **Cluster 29** | 3.23% |
| **Average within Centroid Distance (Cluster 29)** | 0.851 | **Cluster 30** | 1.20% |
| **Average within Centroid Distance (Cluster 30)** | 0.862 | **Cluster 31** | 4.00% |
| **Average within Centroid Distance (Cluster 31)** | 0.842 | **Cluster 32** | 0.77% |
| **Average within Centroid Distance (Cluster 32)** | 0.851 | **Cluster 33** | 0.86% |
| **Average within Centroid Distance (Cluster 33)** | 0.794 | **Cluster 34** | 1.55% |
| **Average within Centroid Distance (Cluster 34)** | 0.788 | **Cluster 35** | 1.33% |
| **Average within Centroid Distance (Cluster 35)** | 0.756 | **Cluster 36** | 0.56% |
| **Average within Centroid Distance (Cluster 36)** | 0.812 | **Cluster 37** | 1.33% |
| **Average within Centroid Distance (Cluster 37)** | 0.832 | **Cluster 38** | 1.55% |
| **Average within Centroid Distance (Cluster 38)** | 0.534 | **Cluster 39** | 3.14% |
| **Average within Centroid Distance (Cluster 39)** | 0.845 | **Cluster 40** | 2.41% |
| **Average within Centroid Distance (Cluster 40)** | 0.860 | **Cluster 41** | 1.63% |
| **Average within Centroid Distance (Cluster 41)** | 0.681 | **Cluster 42** | 1.46% |
| **Average within Centroid Distance (Cluster 42)** | 0.846 | **Cluster 43** | 1.63% |
| **Average within Centroid Distance (Cluster 43)** | 0.847 | **Cluster 44** | 1.24% |
| **Average within Centroid Distance (Cluster 44)** | 0.868 | **Cluster 45** | 1.59% |
| **Average within Centroid Distance (Cluster 45)** | 0.790 | **Cluster 46** | 3.87% |
| **Average within Centroid Distance (Cluster 46)** | 0.812 | **Cluster 47** | 1.55% |
| **Average within Centroid Distance (Cluster 47)** | 0.820 | **Cluster 48** | 0.21% |
| **Average within Centroid Distance (Cluster 48)** | 0.679 | **Cluster 49** | 0.81% |
| **Average within Centroid Distance (Cluster 49)** | 0.829 | | |
| **Davies Bouldin Index** | 5.096 | | |

From the above tables, we conclude that K = 50 yields us the best possible cluster model. K with the value of 50 gives us the least value of Davies Bouldin Index, which is desirable since it signifies low intra-cluster distances and high inter-cluster distances. Moreover, the clusters are decently spread out in terms of cluster sizes and the average within centroid distance of the clusters are close enough.

Hence, with the process of clustering, we obtain 50 as our ‘Best’ number of clusters.