Project Report

Anonymous Author(s)

1 INTRODUCTION

The goal of this project is to study label distributions. When given an instance of a distribution, label distributions assign set of labels to the distribution in preference to assigning a single or multiple labels to that distribution. Label distributions solve the problem of label ambiguity. A trained classification model was provided that divided a message/tweet into different classes indicating the category where this message belongs. The model was trained based on the different labels assigned to each message/tweet. Then a graph was build for the trained model and the graph was then studied using gephi software. To test the model a dataset was chosen that contained 484 tweets that are maybe job related and set of labels are assigned to each of these tweet. Some of the labels are 'coming home from work', 'not job related', 'quitting a job', 'going to work', 'none of the above but job related', etc. The next step was to manually annotate given employment transitions into 12 label choices where the value 1 indicates that a particular employment transition belongs to that label category. After annotation was done, some visualizations were done by computing Cohen kappa's inter-annotator and confusion matrix between two different annotated sheets for each label or category. The team discussed and understood each other's thinking behind how they labelled data. Disagreements were discussed and tweets were relabeled if felt necessary. On the relabeled spreadsheets, each tweet was assigned with a cluster by using the Kmeans clustering algorithm. After preprocessing the text in the messages, from each cluster, top 10 words were identified based on their frequency count.
This entire process was repeated on the data after adding additional labels and the results obtained were compared and analysed with the previous results.

2 METHODS

2.1 Apriori Algorithm

Apriori Algorithm was first founded in 1997 by R. Agrawal and R. Srikant and its purpose is to find frequent itemsets along with association rule learning from a dataset. Apriori algorithm uses a bottom up approach along with breadth first search and hash tree structure to find the itemsets association efficiently. It is done by finding the frequent subsets one at a time along and grouping the candidates which are tested against the data [9].

2.1.1 Parameters: The Apriori algorithm have three parameters [2] that are as below:

(1) ***Support*** : It shows the frequency of occurring an itemset in the dataset. Suppose an item is labelled as x, then support is calculated as the ratio of count of records containing x by the total records that are in the database.
(2) ***Confidence*** : Confidence can be measure of the summation of number of times item y is bought when an item x is bought first. It is calculated as the ratio of support of (xUy) by support of x.
(3) ***Lift*** : Lift is the ratio of confidence of a rule to the expected confidence of that rule.

2.2 Kmeans Clustering Algorithm

Given a set of observations, the Kmeans algorithm divides the observations into clusters. Each entry is assigned with a cluster on the basis of its proximity to a cluster with the nearest mean. This algorithms was first proposed by Stuart Lloyd of Bell Labs in 1957, but it was not officially published as a journal until 1982 [8]. However, essentially the same algorithm was published by Edward W. Forgy in 1965 [7]. Hence this algorithm is sometimes called the Lloyd–Forgy algorithm.

2.2.1 Mathematical Model: Kmean clustering algorithm can be viewed as a minimization problem which consists of two iterative steps. Given an observation:

(1) identify the cluster it belongs to such that the euclidean distance between the observation and the centroid of the cluster is minimum.
(2) re-compute the centroid of the modified cluster.

The algorithm essentially aims to maintain clusters such that the pairwise squared deviations of points in the same cluster are minimized. This can be represented by the following mathematical formula [10]. Here (x1, x2, ..., xn) represent the set of observations, S = S1, S2, ..., Sk represent the set of clusters and μ_i represents the mean of the i^{th} cluster:

$$argminS \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \|^2 \quad (1)$$

2.2.2 Parameters: The kmeans algorithm consists of the following parameters:

(1) ***n_clusters*** : this is an integer that denotes the number of clusters to be formed by the model.For our implementation we used 4 clusters.
(2) ***init*** : this represents method of initialization of the clusters. It can be 'k-means++' or 'random'. 'k-means++' initializes clusters in a smart way such that the clusters are relevant to the data. 'random' initializes clusters in a random manner. For our implementation, we chose *init = 'k-means++'*.
(3) ***n_init*** : this integer value represents the number of times the k-means algorithm will be initialised and run. The default value is set to 10. For our implementation we used the default value.
(4) ***max_iter*** : this integer value represents the maximum number of iterations during each run.The default value is set to 300. For our implementation we used the default value.
(5) ***tol*** : this float value represents the relative tolerance with respect to the Frobenius norm of the difference in the cluster centers of two consecutive iterations in order to declare convergence. The default value is set to 1e-4. For our implementation we used the default value.

(6) ***verbose*** : this integer value represents the verbosity mode of the the model. The default value is set to 0. For our implementation we used the default value.
(7) ***random_state*** : this parameter represents the method of random number generation for initializing clusters. It can either be RandomState instance or None. For our implementation we used None as the clusters were initialized using 'k-means++'.
(8) ***copy_x*** : when this parameter is True, the original data is not modified. Else, the the original data is modified before the function returns.
(9) ***algorithm*** : this parameter represents the variation of the kmeans algorithm to be used. The value of this parameter can be "auto", "full"or "elkan". For our implementation we used the default value i.e "auto".

2.3 Frequent Words per Cluster:

In order to find the frequent words per cluster, the tweets belonging to each cluster were first combined into a single list. The list of tweets was then preprocessed by performing, stemming, lemmatizarion, removing stopwords. The tweets were returned as a bag of preprocessed words. Using the Counter class from the collections module the count for occurrence of each word was then computed. From the above, computed count, the 10 most frequent words from each cluster were derived.

3 RESULT

3.1 Analysis of graph1

For the graph shown below (1) is for the trained model and it has 757 nodes, 12462 edges, and 2 connected components.

The histogram for of the degree (3) and size (2) of connected components is shown as below:

3.2 Analysis of graph2

For the graph shown below (4) is for the trained model and it has 748 nodes, 5592 edges, and 12 connected components.

The histogram for of the degree (6) and size (5) of connected components is shown as below:

3.3 Additional statistical analysis of both graphs

Some of the additional statistical analysis on the graph is stated as below:

(1) Density in a graph is the ratio between the actual number of edges present in the graph to the all possible edges in a graph. Density helps to understand that how closely the network or graph is related. The density of the graph1 is 0.04355119414564593 and density of graph2 is 0.020015892446792517.
(2) Triadic closure depicts the probability of the edges to form a triangle in the graph. Example of this is that if node 1 and node 2 are connected by an edge and node 2 and node 3 are connected by an edge, then triadic closure is the likelihood of node 3 and node 1 to be connected by an edge and hence forming a triangle. The triadic closure of the graph1 is 0.3761279059361714 where triadic closure of graph2 is 0 that is there are no edges forming a triangle
(3) A degree in any graph for a node is the sum of the edges in that node. It helps to understand the most important nodes present in any graph. The degree value for this graph ranges in between 1 to 126. The top 10 nodes according to the maximum degree for graph1 are:

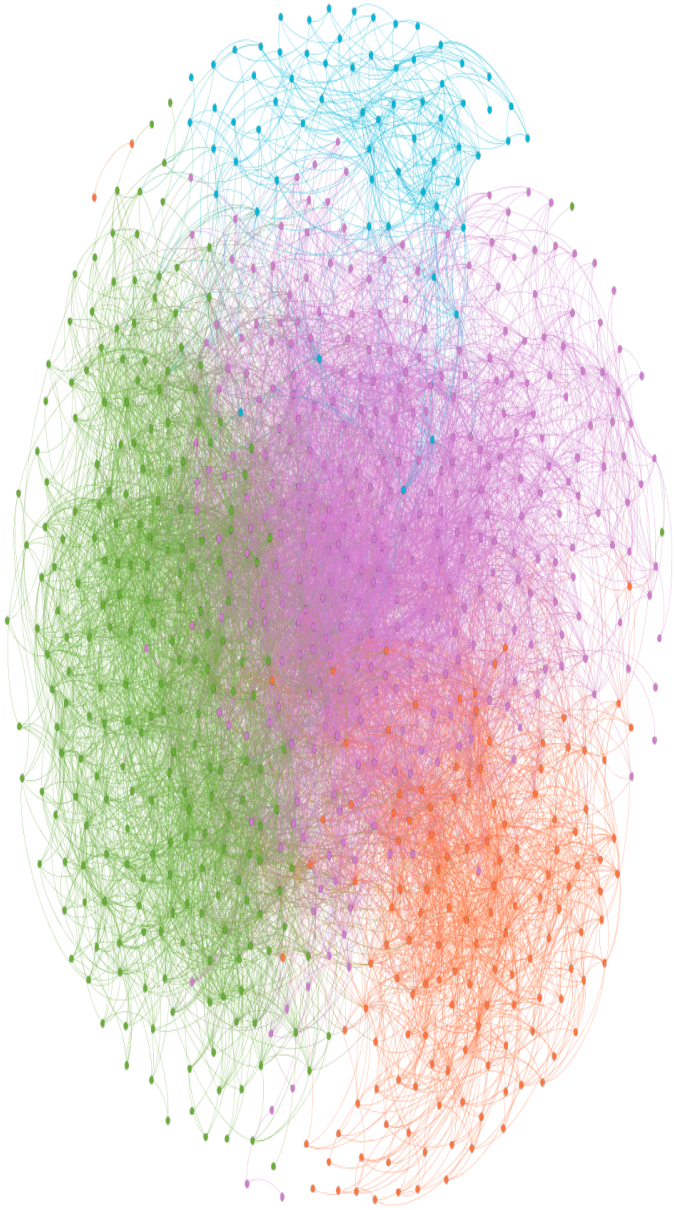

Figure 1: Colored graph1

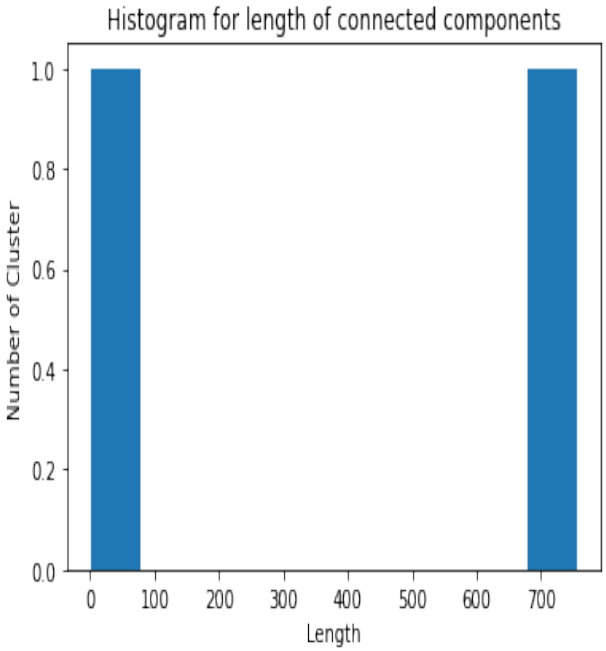

Figure 2: Size of connected components of graph1

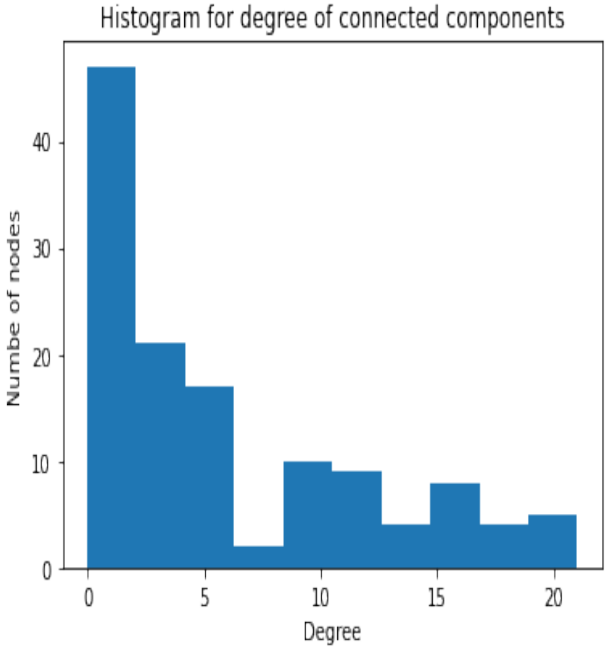

Figure 3: Degree of the graph1

- Node: (0, 1, 0, 0, 1, 0, 2, 0, 4, 2, 1, 0), Degree: 126
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 8, 2, 0, 0), Degree: 121
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 7, 3, 0, 0), Degree: 118
- Node: (1, 1, 0, 0, 0, 0, 1, 0, 5, 1, 1, 0), Degree: 117

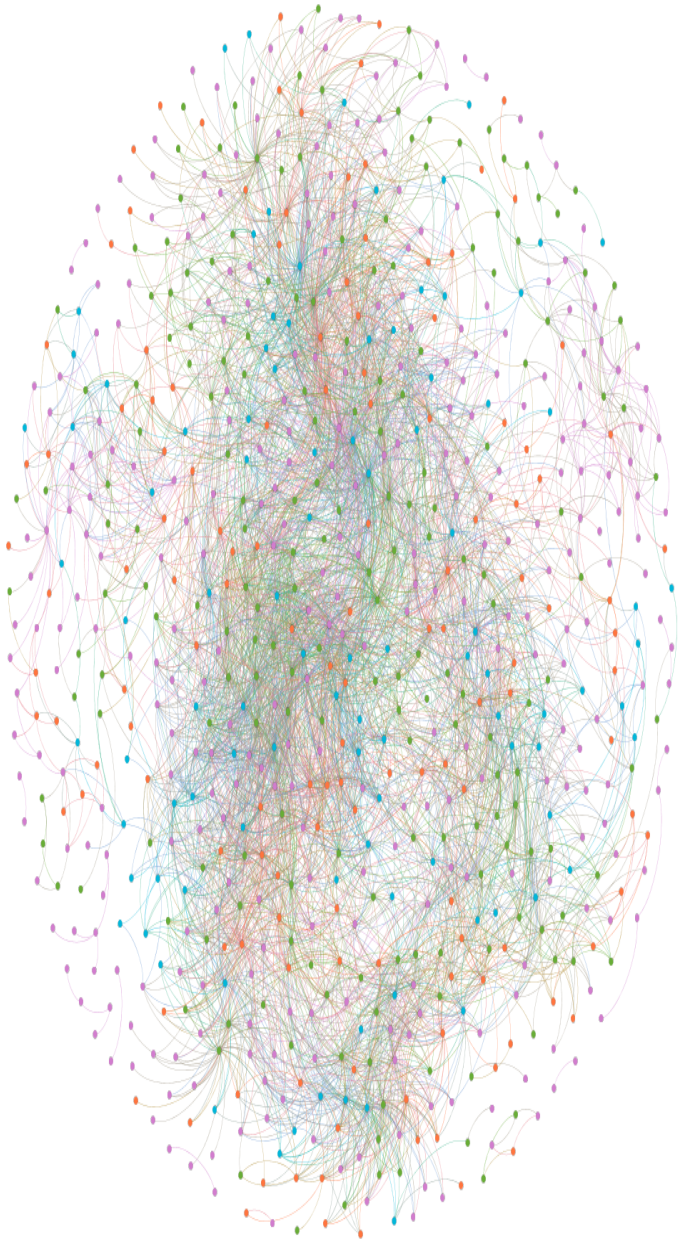

Figure 4: Colored graph2

- Node: (1, 2, 1, 0, 0, 0, 1, 0, 3, 2, 0, 0), Degree: 116
- Node: (1, 2, 0, 0, 0, 0, 2, 0, 3, 2, 0, 0), Degree: 108
- Node: (0, 2, 0, 0, 0, 0, 0, 0, 8, 0, 0, 0), Degree: 106
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 6, 4, 0, 0), Degree: 101
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 10, 0, 0, 0), Degree: 101
- Node: (0, 3, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0), Degree: 101

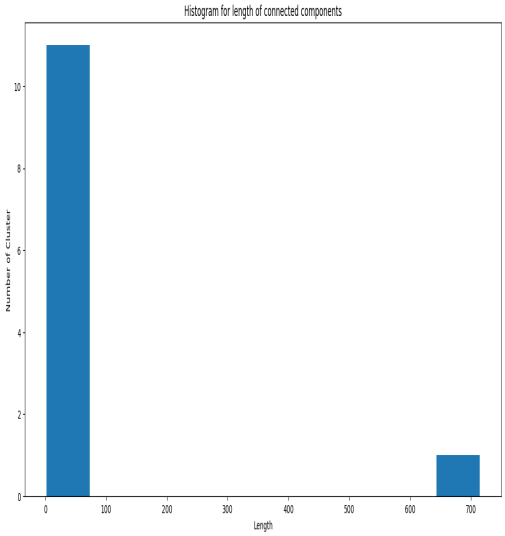

Figure 5: Size of connected components of graph2

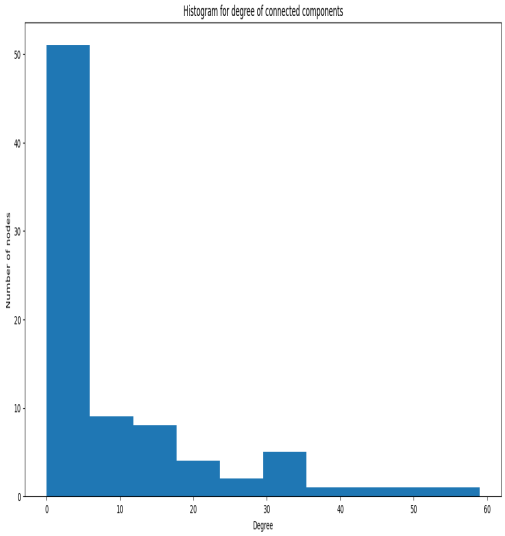

Figure 6: Degree of the graph2

The top 10 nodes according to maximum degree for graph2 are:

- Node: (0, 0, 0, 0, 0, 0, 0, 0, 7, 3, 0, 0), Degree: 82
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 3, 7, 0, 0), Degree: 82
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 8, 2, 0, 0), Degree: 79
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 6, 4, 0, 0), Degree: 79
- Node: (1.0, 3.0, 1.0, 0.0, 4.0, 0.0, 2.0, 1.0, 10.0, 2.0, 1.0, 0.0), Degree: 79
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 5, 5, 0, 0), Degree: 78
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 4, 6, 0, 0), Degree: 77
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 2, 8, 0, 0), Degree: 71
- Node: (0, 4, 0, 0, 0, 0, 0, 0, 4, 2, 0, 0), Degree: 62
- Node: (0, 0, 0, 0, 0, 0, 0, 0, 9, 1, 0, 0), Degree: 62

3.4 Visualization

To visualize the annotations between two annotated sheets, two metrics namely Covariance matrix and Cohen's kappa inter-annotator were used for each label or category in the dataset. Some of the visualizations for few labels have been attached below along with its cohen's value.

- For label 'Coming home from work' the cohen's kappa value is 0.9157865009884214 and its covariance matrix is shown in figure (7)

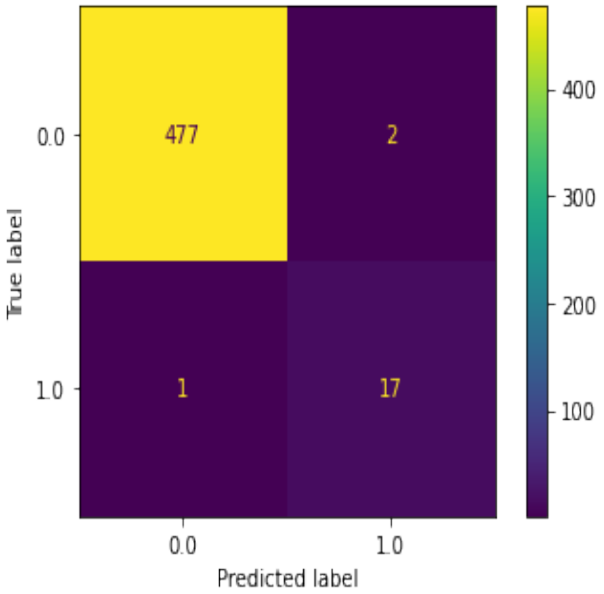

Figure 7: Covariance Matrix for label 'coming home from work'

- For label 'Complaining about work' the cohen's kappa value is 0.9802409255357214 and its covariance matrix is shown in figure (8)
- For label 'Getting fired' the cohen's kappa value is 0.6657700067249496 and its covariance matrix is shown in figure (9)
- For label 'Getting hired/job seeking' the cohen's kappa value is 0.9844264093002851 and its covariance matrix is shown in figure (10)
- For label 'Going to work' the cohen's kappa value is 0.8792956891317547 and its covariance matrix is shown in figure (11)
- For label 'None of the above but job related' the cohen's kappa value is 0.9239060538322565 and its covariance matrix is shown in figure (12)

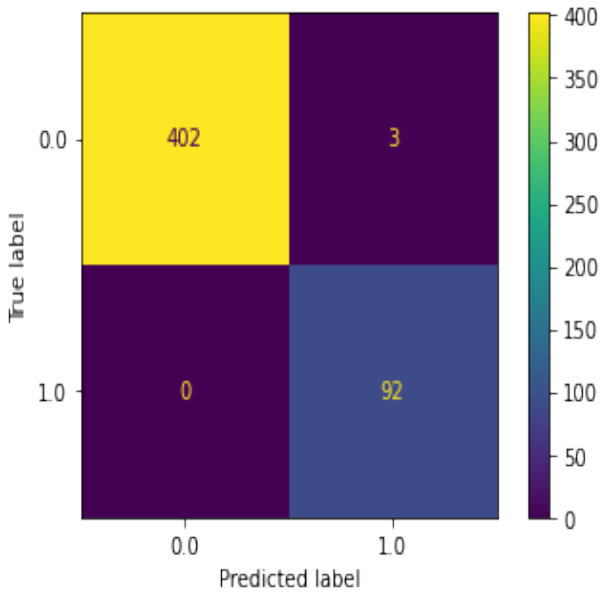

Figure 8: Covariance Matrix for label 'complaining about work'

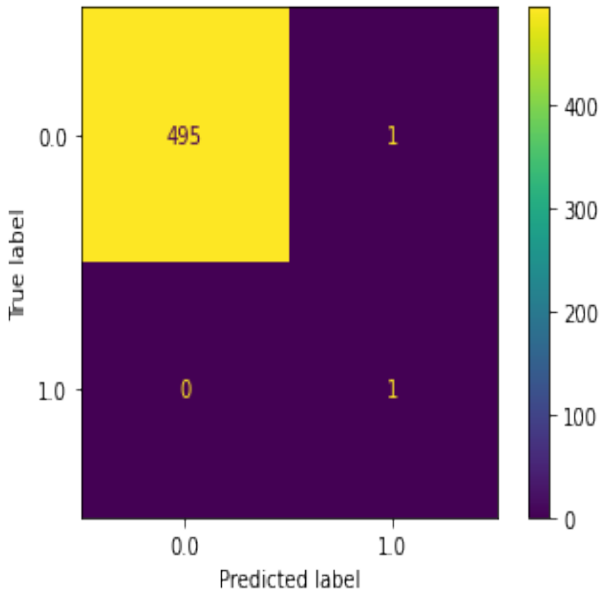

Figure 9: Covariance Matrix for label 'getting fired'

- For label 'Not job related' the cohen's kappa value is 0.9577400916434409 and its covariance matrix is shown in figure (13)
- For label 'Offering support' the cohen's kappa value is 0.9619302949061662 and its covariance matrix is shown in figure (14)

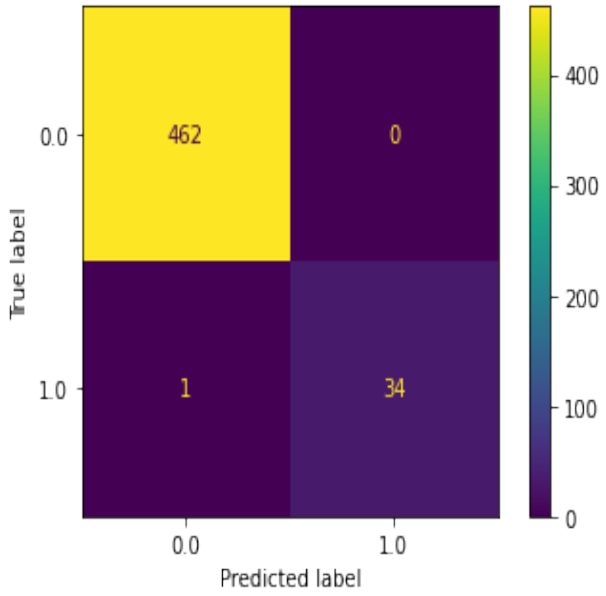

Figure 10: Covariance Matrix for label 'getting hired/job seeking'

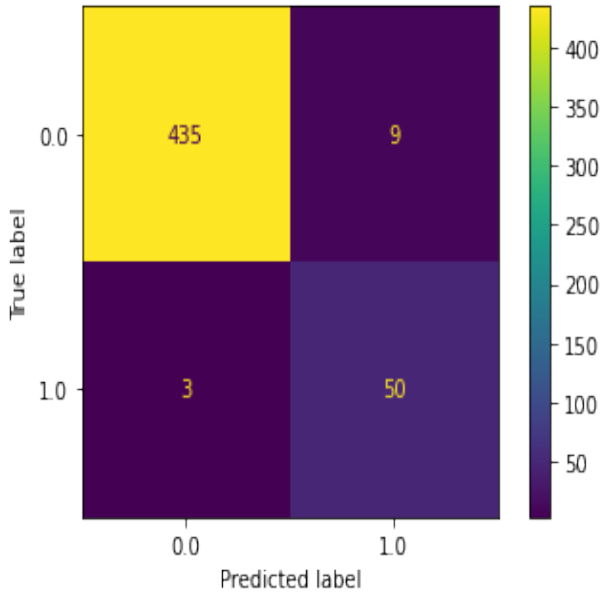

Figure 11: Covariance Matrix for label 'going to work'

3.5 Frequent Words per Cluster

The tweets per cluster were first concatenated into a single string and this string was preprocessed using the Python nltk library [1]. The steps in preprocessing invloved lemmatization using the WordNetLemmatizer class from the nltk library [6], stemming using the PorterStemmer class from the nltk library [5] and removal of stopwords using gensim.parsing.preprocessing.STOPWORDS [4].

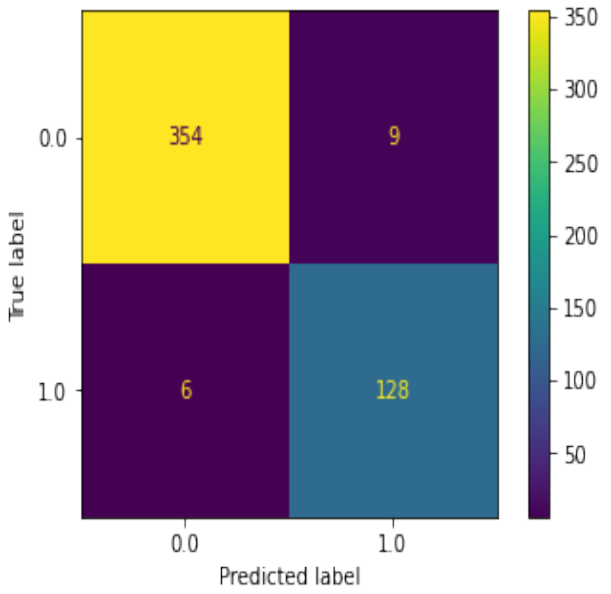

Figure 12: Covariance Matrix for label 'None of the above but job related'

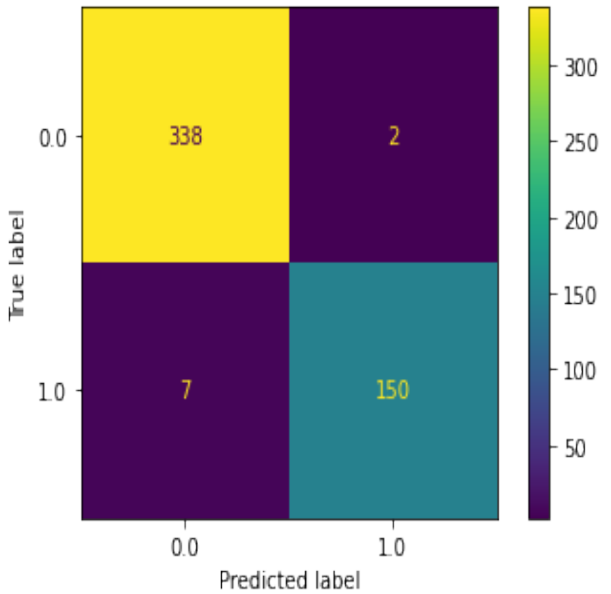

Figure 13: Covariance Matrix for label 'Not job related'

For the bag of words obtained after preprocessing, the count for each unique word was derived using the Counter class from the collections library in Python [3] which creates a dictionary consisting of the words as keys and their counts as values. From this dictionary, 10 most frequent words for each cluster were obtained.

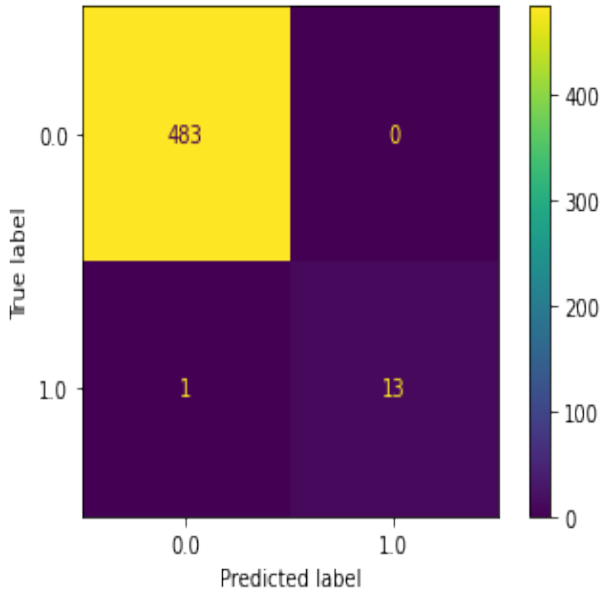

Figure 14: Covariance Matrix for label 'Offering support'

Word	Count
work	71
http	19
like	11
today	11
go	8
time	6
love	6
best	6
school	6
road	6

Table 1: Top 10 words in cluster 1:

Word	Count
work	95
today	12
like	11
time	9
go	9
come	8
shift	7
gonna	7
hour	6
sleep	6

Table 2: Top 10 words in cluster 2

Word	Count
work	60
go	9
like	7
today	6
readi	6
come	5
tomorrow	5
want	5
morn	4
get	4

Table 3: Top 10 words in cluster 3

Word	Count
work	57
go	8
like	7
tomorrow	7
need	6
start	6
good	6
http	5
come	5
home	5

Table 4: Top 10 words in cluster 4

Clusters after adding labels:

Word	Count
work	866
like	111
today	111
go	103
http	84
come	70
time	66
want	57
good	48
tomorrow	45

Table 5: Top 10 words in cluster 1:

Word	Count
work	44
http	16
like	8
come	6
best	6
road	6
bridg	6
love	5
go	5
want	5

Table 6: Top 10 words in cluster 2

Word	Count
work	78
today	17
like	13
go	11
http	9
gonna	7
school	6
come	6
time	6
tomorrow	6

Table 7: Top 10 words in cluster 3

Word	Count
work	140
go	17
toady	12
time	10
like	10
come	9
hour	9
shift	9
want	8
readi	8

Table 8: Top 10 words in cluster 4

4 DISCUSSION

4.1 Disagreement of Tweets

In the two annotation sheets we had total 24 disagreements. After discussing briefly on these disagreements, 14 disagreements were changed or were relabelled and the rest of the 10 disagreements were kept as it is. Some of the disagreements that were changed in both of our files are as below:

- The item id is 437073796059889664 and the message is '@SOMEONE I work at journeys!'. For this particular message one of the member labelled it as 'Not job related' because according to them the person was just suggesting the other person that they work at a particular place and it is nowhere mentioned that its related to their work where the other person labelled as 'None of the above but job related' because it was somehow related to work as they were suggesting that somehow the person is discussing about work. So for this the label was changed to 'None of the above but job related'.
- The item id is 572752474295148544 and the message is 'Imma b mad if dey don't come b4 I go to work .. I had sum errands to run but put them off to sit home for dis delivery ..'. For this message one member labelled it as 'going to work' because the person in this message is talking to about doing some errands before going to work while the other member labelled it as 'not job related' as they thought that the person is talking about being angry if others don't show up before the person goes to work. So for this the label was changed to 'Going to work'.

10 of the disagreements were not changed as both the team members thought that their labels were relevant according to their perspective and some of the examples are as below:

- The item id is 530789477942558720 and the message is 'All I wanna wear to work is a hoodie and sweatpants'. For this message one of the member labelled it as 'not job related' because the person in this message is just deciding on what type of clothes they want to wear while going to work and as the person is discussing about the type of clothes for 'work' the other member labelled it under 'None of the above but job related'.
- The item id is 464002326286434304 and the message is 'I think I'm getting sicker not better feel like ass might stop to pick up some stuff on the way to work for assistance Tissues NasalSpray'. For this message one of the member labelled it as 'not job related' because the person in this message is not feeling well and hence they want to grab some stuff from store before going to work while the person labelled it under 'Going to work' because the person in this message is talking about doing some things before they go to work.

4.2 Addition of Labels

By adding the the number of labels, the number of nodes were almost similar, while the number of edges decrease in a significant amount for graph2 than in graph1. Whereas, number of connected components increase to 12 in graph2 as in graph1 where number of connected components are 2. Density of graph also reduces to half while the triadic closure becomes 0 from 0.3761. Increasing the labels while keeping the number of clusters same, made the clustering more cluttered around each centroid which can be expected due to the increased data-points.

REFERENCES

[1] 2021. https://www.nltk.org/
[2] 2021. Apriori Algorithm in R Programming. https://www.geeksforgeeks.org/apriori-algorithm-in-r-programming/
[3] 2021. collections - Container datatypes¶. https://docs.python.org/3/library/collections.html#collections.Counter
[4] 2021. Gensim: topic modelling for humans. https://radimrehurek.com/gensim/parsing/preprocessing.html
[5] 2021. nltk.stem.porter. https://www.nltk.org/_modules/nltk/stem/porter.html
[6] 2021. nltk.stem.wordnet. https://www.nltk.org/_modules/nltk/stem/wordnet.html
[7] E. W. Forgy. 1965. Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics* 21 (1965), 768–769.
[8] S. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137. https://doi.org/10.1109/TIT.1982.1056489
[9] Wikipedia contributors. 2018. Apriori algorithm— Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Apriori_algorithm. [Online; accessed 8-December-2021].
[10] Wikipedia contributors. 2021. K-means clustering — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=K-means_clustering&oldid=1056937903. [Online; accessed 8-December-2021].