



Automatic Ticket Assignment Capstone Project Report

AIML





Index

Index	2
Team Details.....	4
Summary of the problem statement, Data and findings	5
Understanding the Business	5
Risks Involved	5
Background and Objective	6
Data & Findings	6
Data Fields	6
Sample data.....	6
Observations.....	7
Summary of the approach to EDA and Pre-Processing	8
Data Pre-Processing and Cleaning	8
Data cleaning	8
Lemmatization & Stop words removal	8
Data Visualization	9
High Imbalance.....	9
Assignment Group Distribution	10
Word Length Distribution	10
Character Length Distribution	11
Word Cloud After Data Pre-processing and Cleaning	13
Charts	23
Decide Model and Model building	27
Word Embedding.....	27
Text Augmentation	28
Count Vectorization.....	28
Tf idf Vectorization.....	28
Bi-directional LSTM Model.....	28
Machine Learning Models	29
Deep Learning Models.....	32





ML Model Performance	35
DL Model Performance	35
Comparison to benchmark	36
Implications	36
Limitations.....	36
Closing Reflections	36



Team Details

Capstone Project Title AUTOMATIC TICKET ASSIGNMENT

Description

Apply AI techniques to classify incidents to right functional groups and help organizations reduce the resolving time and switch focus on more productive tasks.

Group 10 NLP 1 Members

Mentor

Mr Sandeep Raghuwanshi

Name

Ankit Jain

Sowmiya S

Seema Kumari Singh

Pushpendra

Vinoth Prakash

Email ID

ankit2704smit@gmail.com

ssowmya137@gmail.com

seema.singh71@gmail.com

spushpendra14@gmail.com

prakashvinith16@yahoo.com





Summary of the problem statement, Data and findings

Understanding the Business

IT leverages Incident Management process to ensure there is no disruption to business operations. Any unplanned disruption can cause interruption to business services. Incident management process helps in identification of issues or problems faced by users or operation teams. Manual assignment of incidents is time consuming and requires human intervention. There may be lag in incident resolution due to human errors or if incidents are not routed appropriately. On the other hand, manual assignment also increases the response and resolution times which result in user satisfaction deterioration / poor customer service.

Risks Involved

The manual assignment of these incidents might have below disadvantages:

- ❖ More resource usage and expenses.
- ❖ Human errors - Incidents get assigned to the wrong assignment groups
- ❖ Delay in assigning the tickets
- ❖ More resolution times
- ❖ If a particular ticket takes more time in analysis, other productive tasks get affected for the

Service Desk





Background and Objective

To apply techniques and learnings to make ticket assignment more cost-effective, less resolution time so that service desk team can focus on other productive tasks. We are able to see that the current system is capable of assigning 70+% of the tickets correctly. Our target is to automatically classify tickets and directing them to appropriate groups at the earliest, helps in improving the throughput in the ticketing pipeline of an organization.

Data & Findings

Data format CSV
Total Records 8500

Data Fields

Short description	A brief overview of the issue faced by the user
Description	Detailed description of the issue
Assignment group	GRP_0 ~ GRP_73 (total 74 classes of Assignment group)

Sample data

Short description	Description	Assignment group
login issue	-verified user details.(employee# & manager name) -checked the user name in ad and reset the password. -advised the user to login and check. -caller confirmed that he was able to login. -issue resolved.	GRP_0
Outlook	received from: hmjdrvpb.komuaywn@gmail.com hello team,	GRP_0



Short description	Description	Assignment group
	my meetings/skype meetings etc are not appearing in my outlook calendar, can somebody please advise how to correct this? kind	
cant log in to vpn	received from: eylqgodm.ybqkwiam@gmail.com hi i cannot log on to vpn best	GRP_0

Observations

1. High imbalance seen in data with GRP_0 having 40%+ percent of representation
2. Many groups/classes are with very little representation.
3. Null values:

Short description 8

Description 1

Assignment group 0

4. Very few tickets have non-English descriptions
5. Four columns – Short Description, Description, Caller and Assignment group
6. 74 Assignment groups found - Target classes
7. Caller names in a random fashion (may not be useful for training data)
8. European non-English language also found in the data
9. Email/chat format in description
10. Symbols & other characters in the description
11. Hyperlinks, URLS & few image data found in the description
12. Blanks found either in the short description or description field
13. Few descriptions same as the short description
14. Few words were combined together
15. Spelling mistakes and typo errors are found



Summary of the approach to EDA and Pre-Processing

Data Pre-Processing and Cleaning

Below steps have been performed for initial pre-processing and clean-up of data:

1. Dropped the caller field as the data was not found to be useful for analysis
2. Replaced Null values in short description & description with space.
3. Merged Short Description & Description fields for analysis
4. Contraction words found in the merged Description are removed for ease of word modelling
5. Changed the case sensitivity of words to lower case
6. Removed Hashtags, Hyperlinks, URLs, HTML tags, Emoji and non-ASCII symbols from merged fields.
7. Translating all languages (German) to English
8. Tokenization of merged data
9. Removal of Stop words and Meaningless words
10. Lemmatization
11. WordCloud created
12. Attempted to do spell check
13. Created Plot to understand the distribution of words
14. Removal of line breaks and tabs (`\r\n\t`)
15. Removal of special characters
16. Removal of punctuation
17. Removal of extra spaces
18. Missing value imputation

Data cleaning

Data cleaning has been taken care by splitting the various functions which would remove and clean any unwanted or misleading information.

All words have been converted to lower case.

Header and sender information from the emails have been removed.

All the numbers, non-dictionary characters, newline characters, hashtag, HTML entities, emojis, hyperlinks, extra spaces and unreadable characters have been removed.

We have ensured to remove any caller names included in the description column.

Lemmatization & Stop words removal

Stop words have been removed using nltk corpus modules.

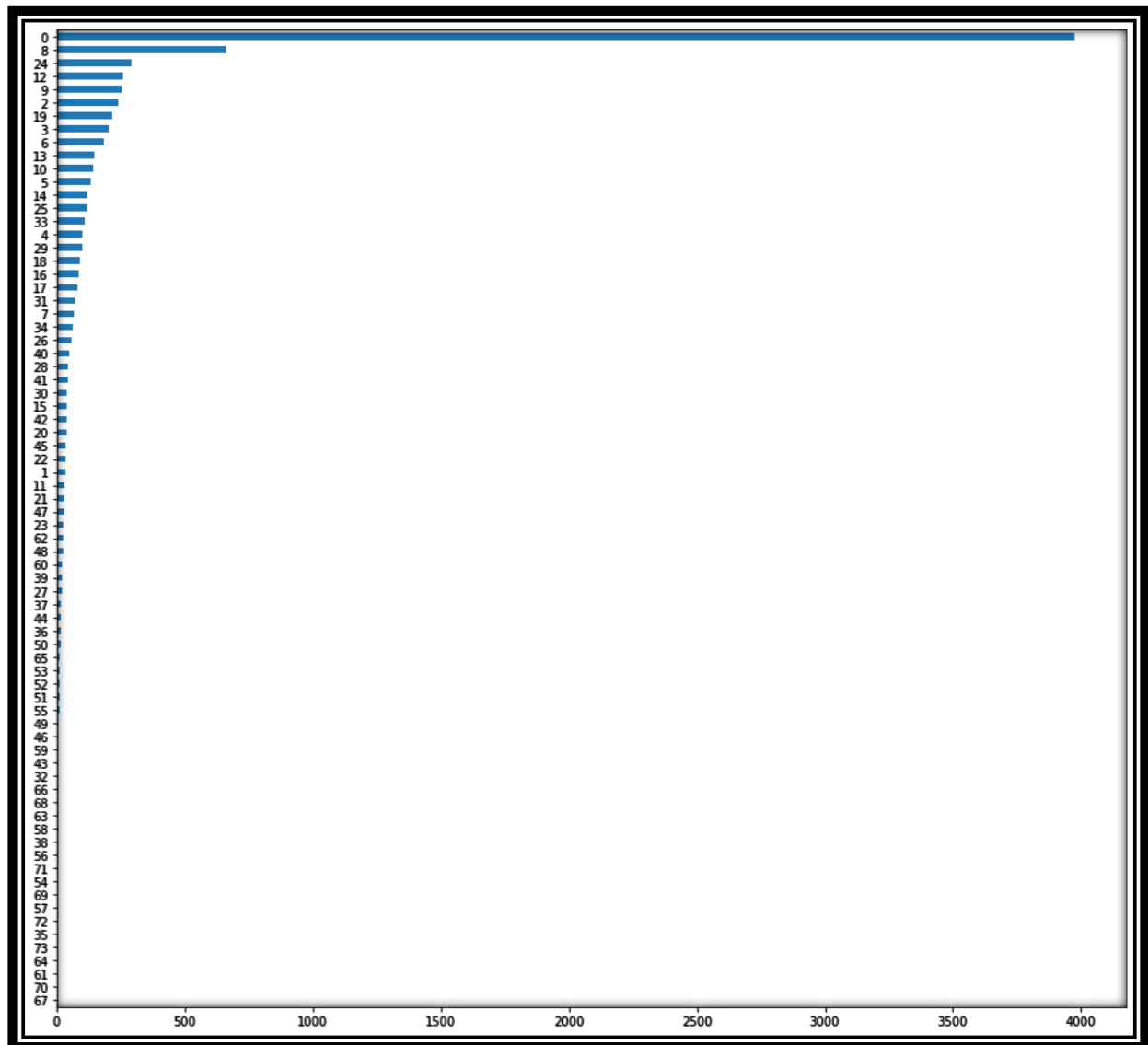
Lemmatization is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. Lemmatization is similar to Stemming but it brings context to the words. So, it links words with similar meanings to one word.

We have preferred Lemmatization over Stemming because lemmatization does morphological analysis of the words

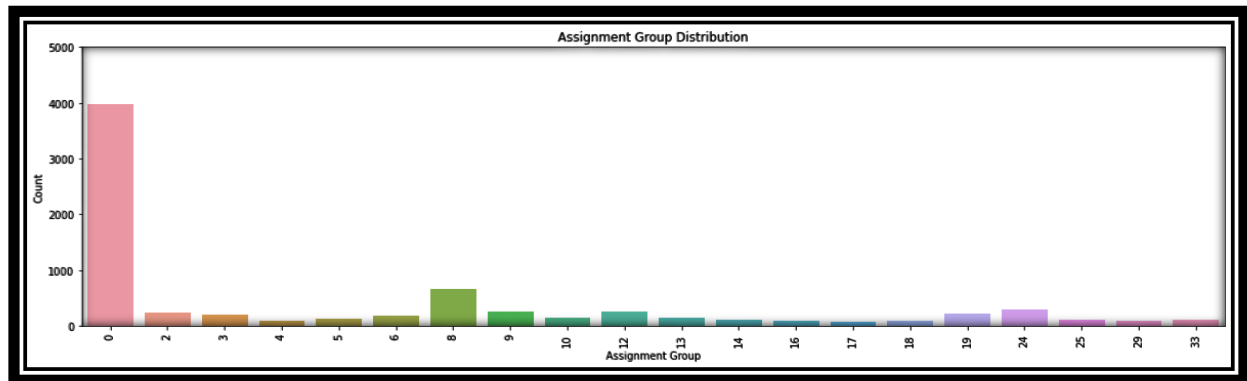


Data Visualization

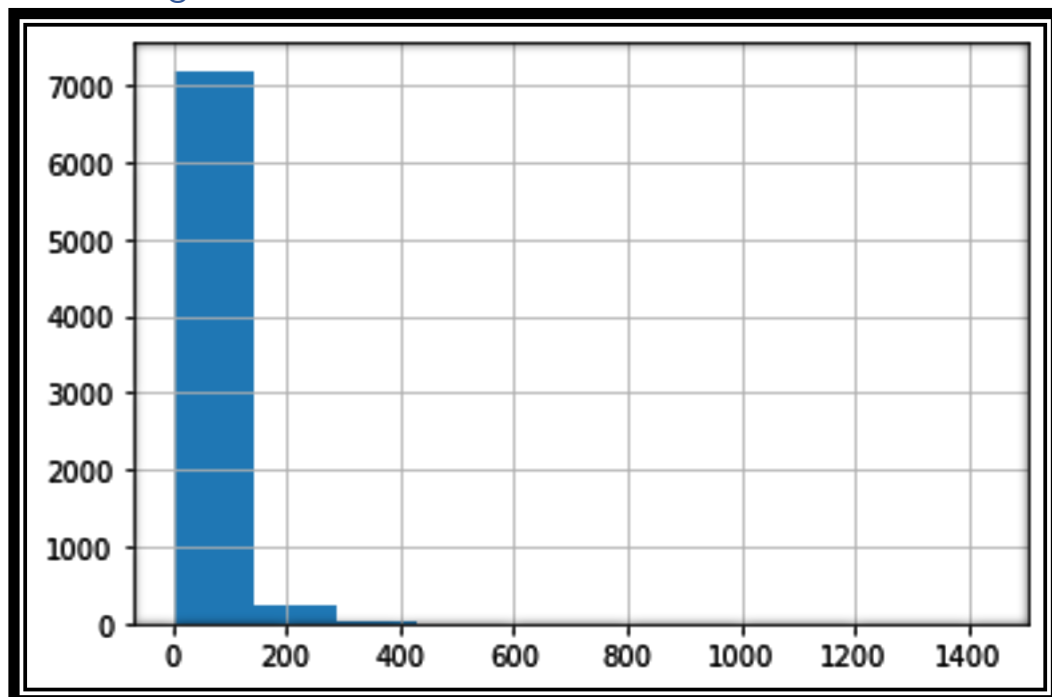
High Imbalance



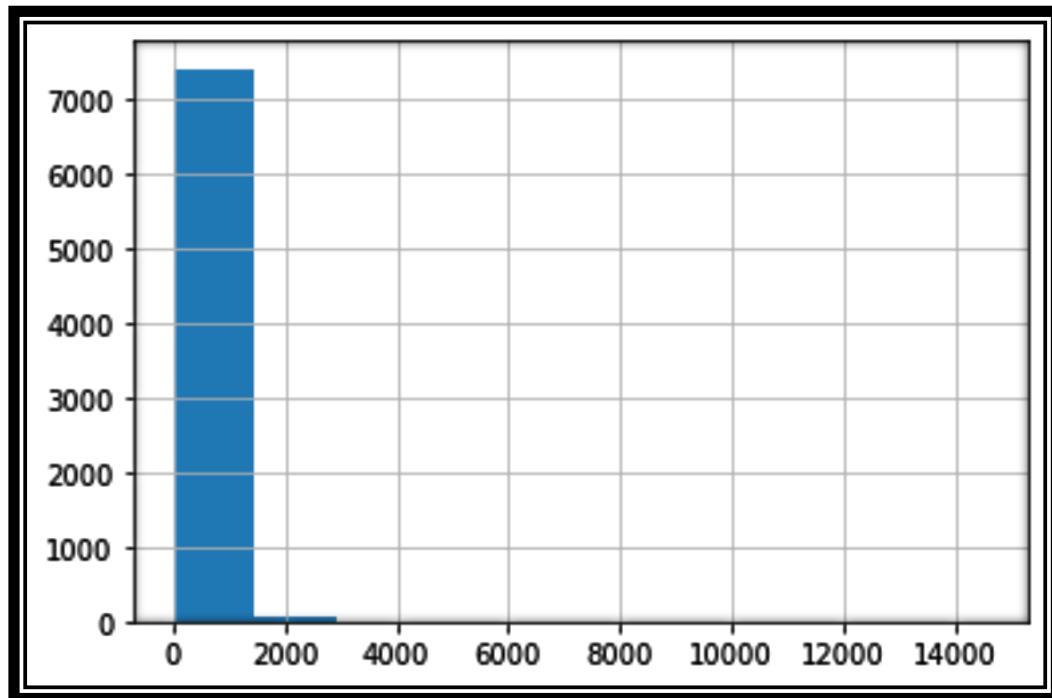
Assignment Group Distribution



Word Length Distribution



Character Length Distribution

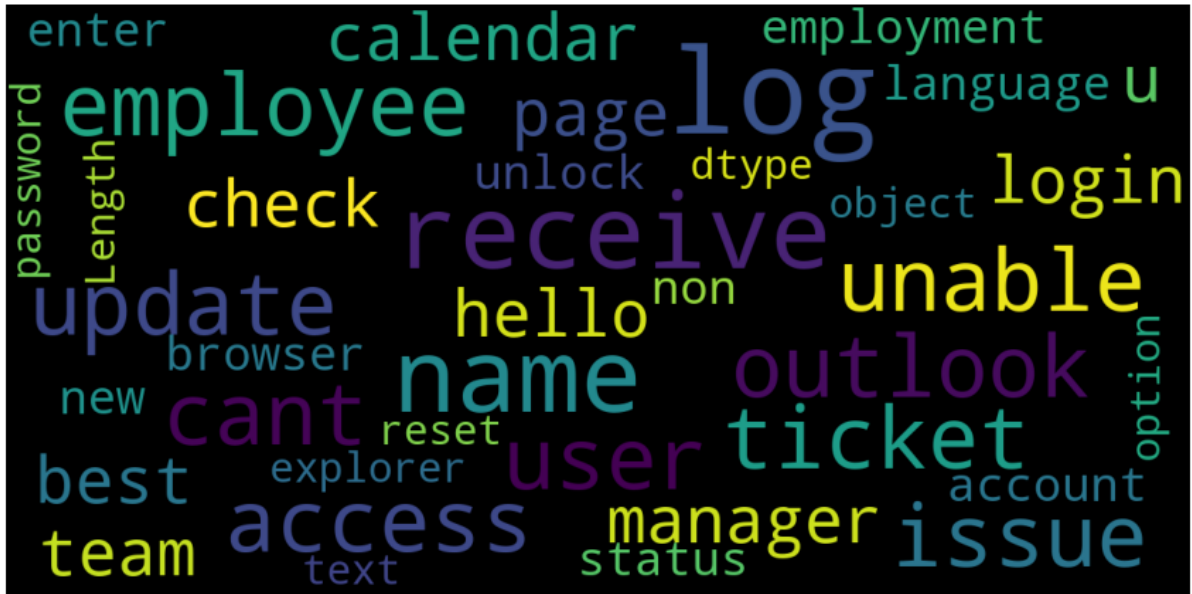


Percentage of Each Group

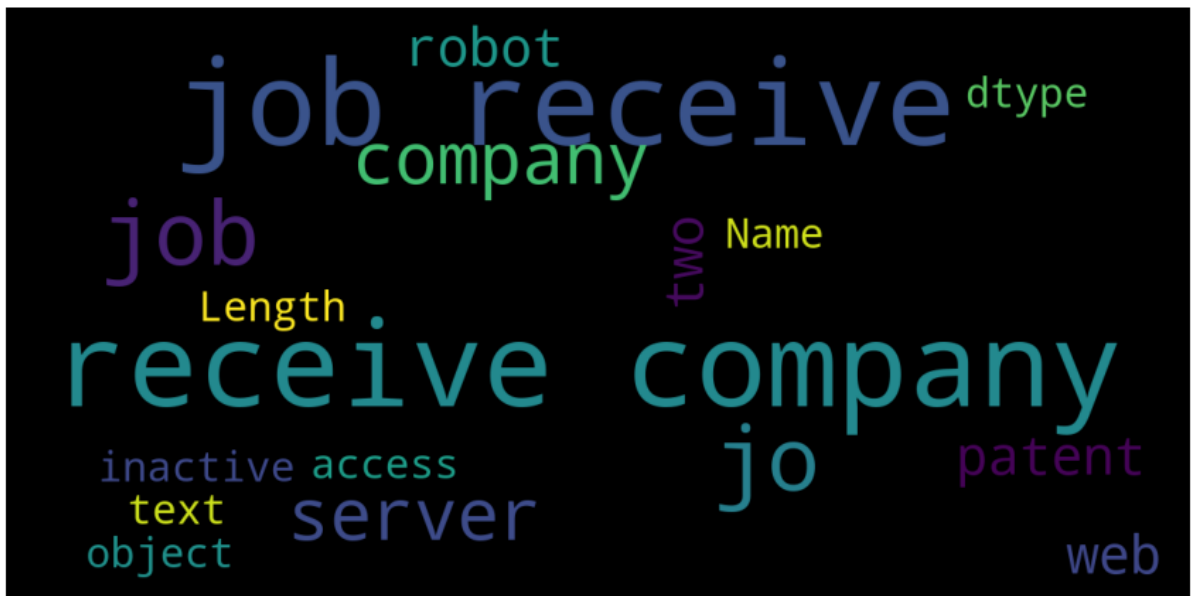
Group	Percentage
0	46.8%
1	0.1%
2	0.1%
3	0.1%
4	0.1%
5	0.1%
6	0.1%
7	0.1%
8	7.8%
9	3.4%
10	0.1%
11	0.1%
12	3.0%
13	0.1%
14	0.1%
15	0.1%
16	0.1%
17	0.1%
18	0.1%
19	0.1%
20	0.1%
21	0.1%
22	0.1%
23	0.1%
24	0.1%
25	0.1%
26	0.1%
27	0.1%
28	0.1%
29	0.1%
30	0.1%
31	0.1%
32	0.1%
33	0.1%
34	0.1%
35	0.1%
36	0.1%
37	0.1%

Word Cloud After Data Pre-processing and Cleaning

Grp_0



Grp_8



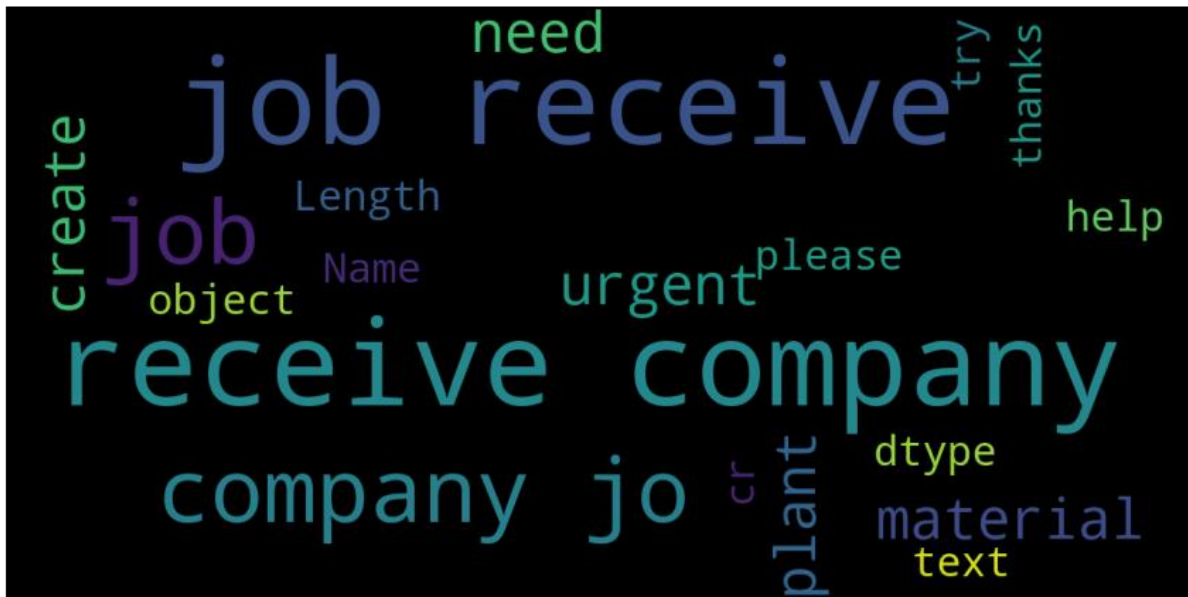
A word cloud of computer-related terms. The most prominent words are 'server', 'receive', 'production', 'volume', 'disk', 'location', 'run', 'access', 'space', 'job', 'host', 'label', 'transition', 'work', 'team', 'provide', 'please', 'system', 'error', 'available', 'live', 'view', 'unable', 'support', 'text', 'memory', 'now', 'internal', 'wrong', 'usage', 'object', 'drive', 'via', 'Name', 'wro', 'number', 'folder', 'tool', 'backup', 'company', 'process', 'infrastructure', 'find', 'dtype', 'hello', 'pr', 'usage', 'object', 'drive', 'via', 'Name', 'wro', 'number', 'folder', 'tool', 'backup', 'company', 'process', 'infrastructure', 'find', 'dtype', 'hello', 'pr'.

A word cloud featuring various terms in different colors and sizes. The most prominent words are 'receive' (large, light blue), 'company' (large, teal), 'job' (medium, purple), and 'analysis' (vertical, purple). Other visible words include 'dtype', 'ess', 'message', 'Length', 'team', 'hello', 'regi', 'screen', 'customer', 'enhance', 'object', 'Name', 'bus', 'error', 'text', 'portal', 'excel', 'field', 'jo', 'group', 'business', 'add', 'higo', and 'file'. The words are scattered across a dark background with a subtle grid pattern.

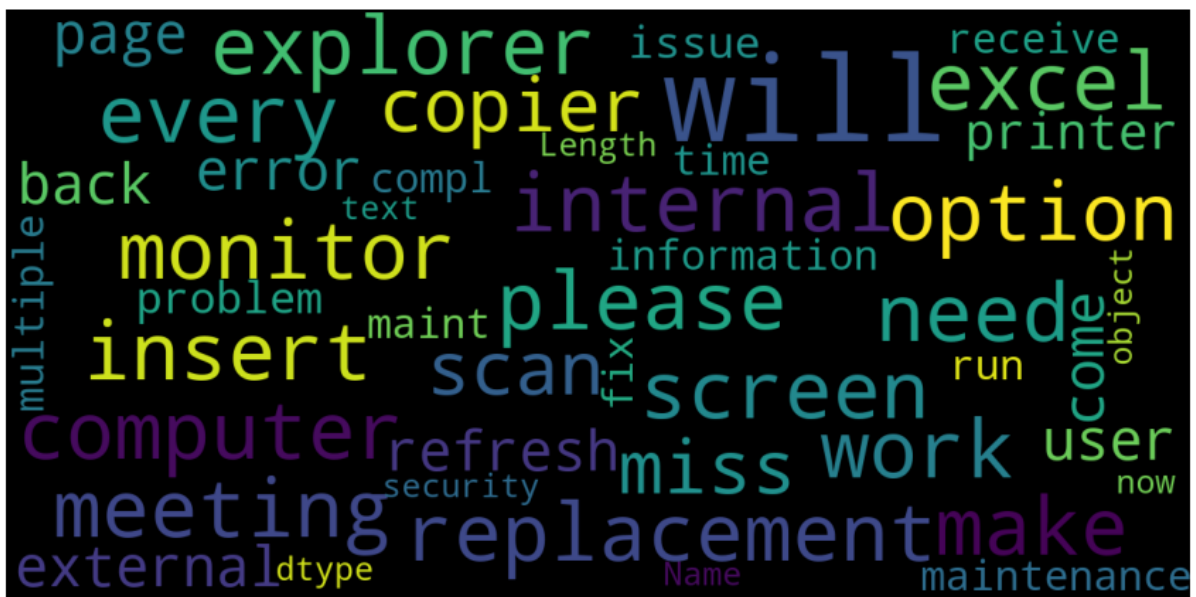


[illegible]

Grp_06

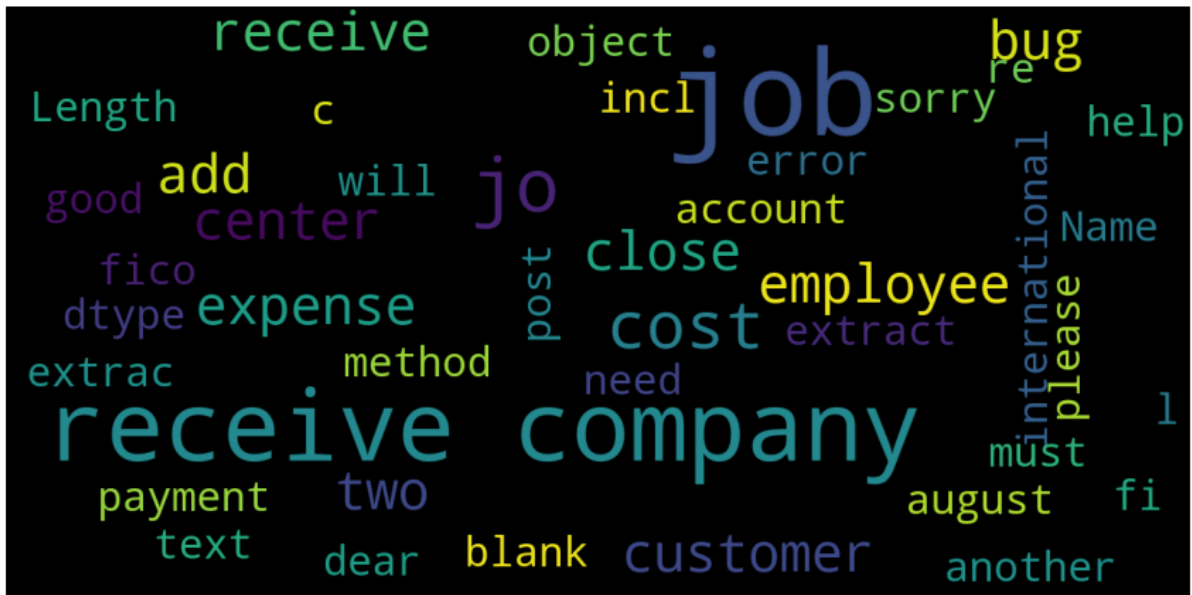


Grp_03

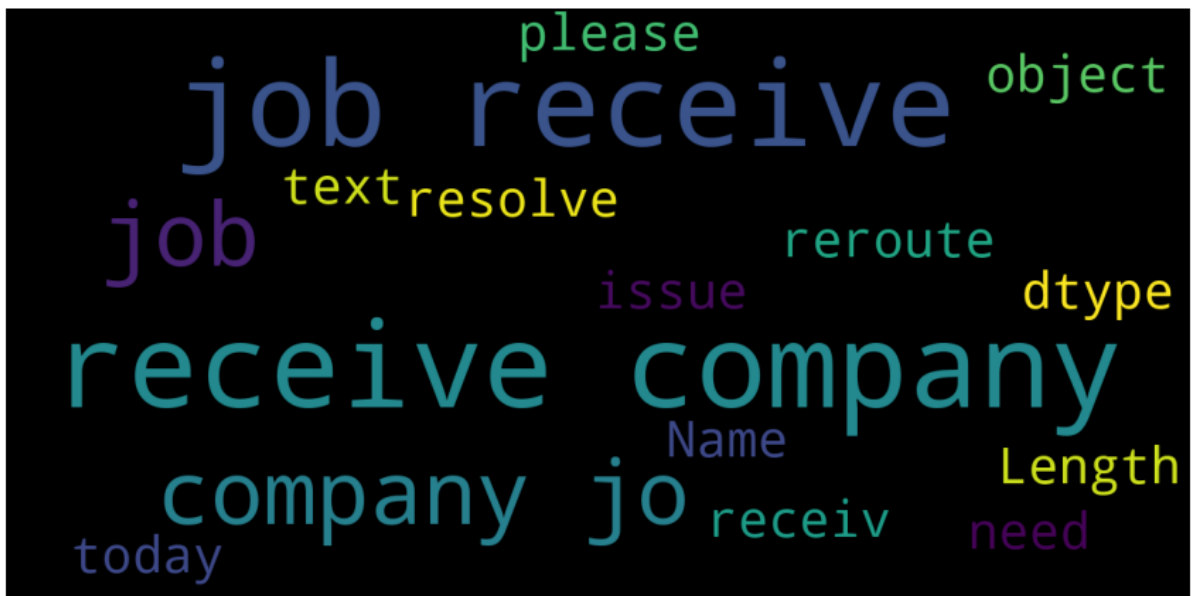


[illegible]

Grp_10



Grp_05



[illegible]

A word cloud visualization of the top 100 words from the 'nlp' dataset. The words are arranged in a dense, overlapping manner, with colors ranging from green to purple. The most prominent words are 'network', 'interface', 'controller', 'server', 'core', 'phone', 'receive', 'address', 'outage', and 'try'.

[illegible]

A word cloud visualization of words from a dataset. The words are arranged in a circular pattern, with larger words indicating higher frequency. The most prominent words include "receive", "subject", "line", "check", "help", "need", "urgent", "send", "request", "delete", "work", "good", "please", "trouble", "day", "now", "add", "note", "correctly", "august", "assist", "creation", "delivery", "de", "length", "will", "warehouse", "dtype", "pick", "aw", "value", "document", "t", "re", "account", "one", "see", "project", "look", "release", "Name", "text", "double", "shipment", "customer", "dear", "object", "print", "fix", "hi", "s", "p", "s", "rout", "urgently", "creation".

A word cloud of terms related to password security. The most prominent word is 'password' in large blue font. Other significant words include 'reset' in large blue font, 'unlock' in large yellow font, 'change' in large yellow font, 'instruction' in large blue font, 'system' in large yellow font, 'error' in large yellow font, 'object' in large blue font, 'employee' in large green font, 'please' in large blue font, 'new' in large blue font, 'per' in large purple font, 'Length' in large blue font, 'dtype' in large green font, 'text' in large green font, 'user' in large purple font, '1' in large blue font, 'hi' in large blue font, 'V' in large purple font, 'Name' in large blue font, 'account' in large purple font, 's' in large yellow font, 'instruction' in large blue font, 'change' in large yellow font, and 'hi' in large blue font. The words are scattered across the image with varying colors and sizes.



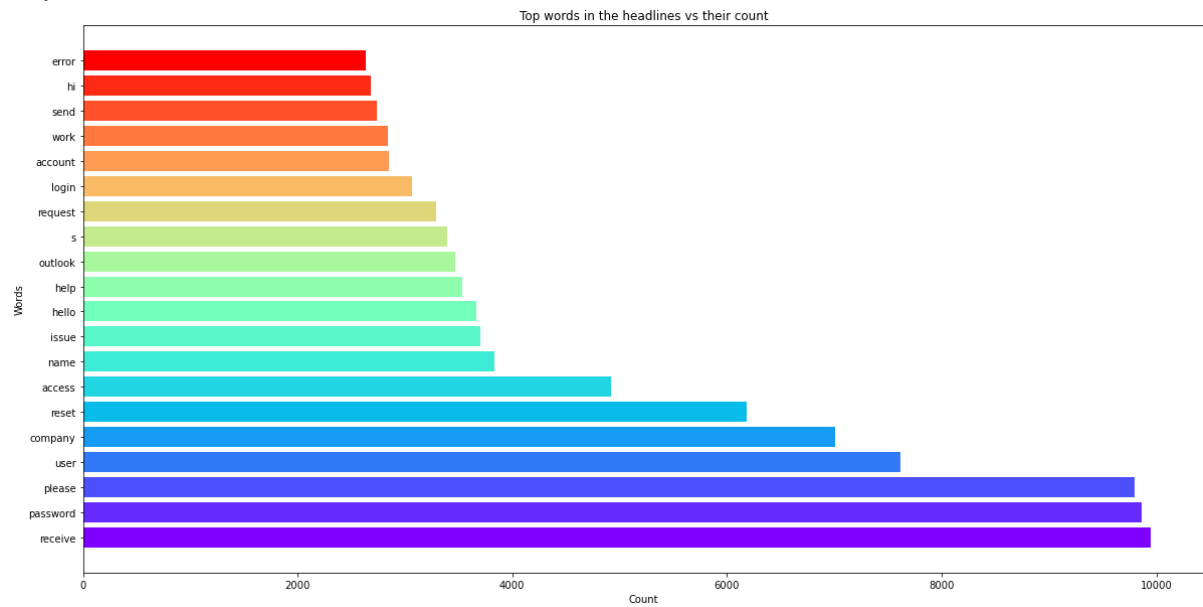




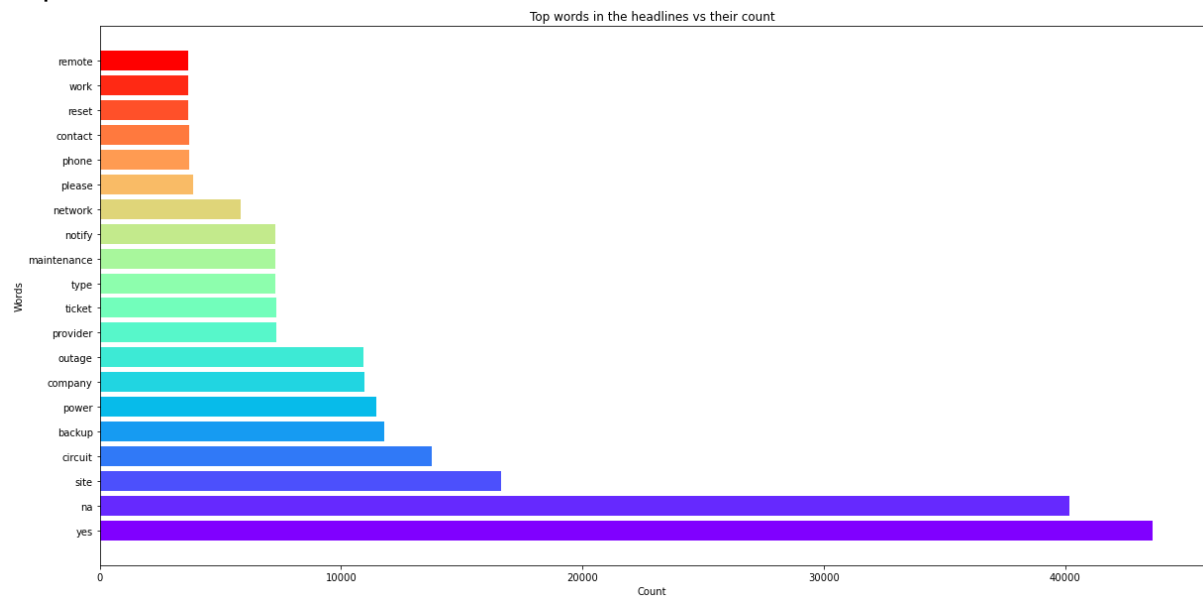
Charts

Frequent words used in Top 20 groups [Sharing charts for a few of them]

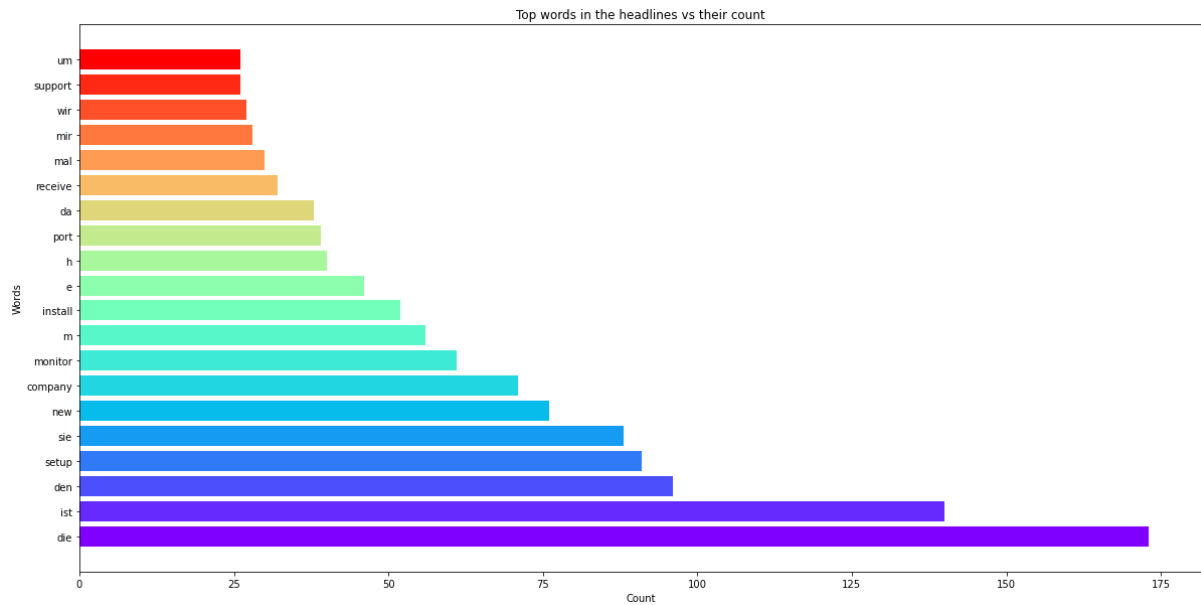
Grp_0



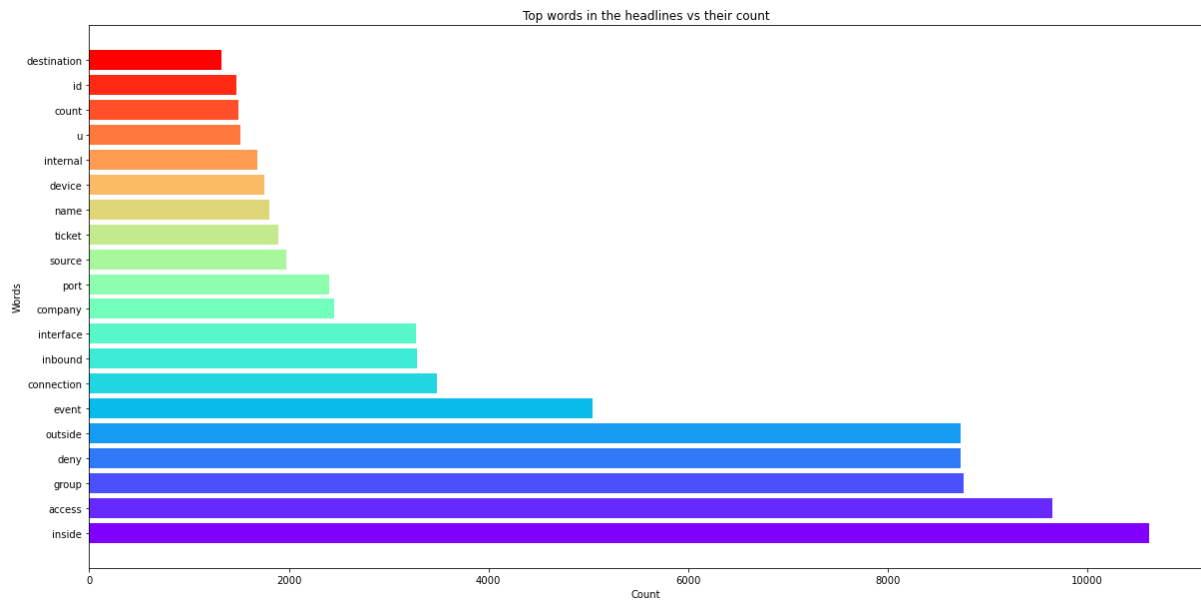
Grp_8



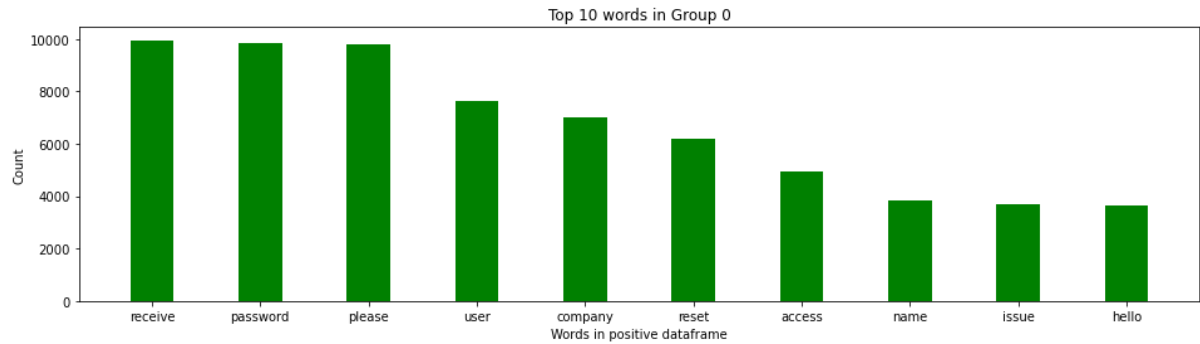
Grp_24



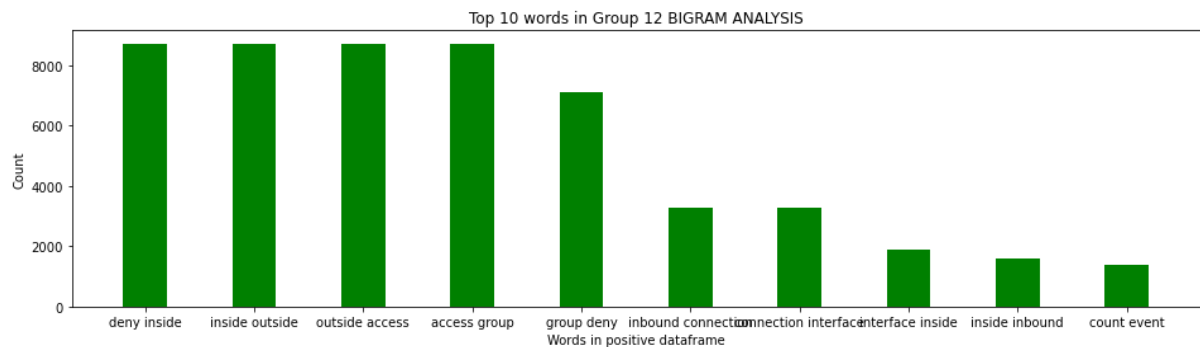
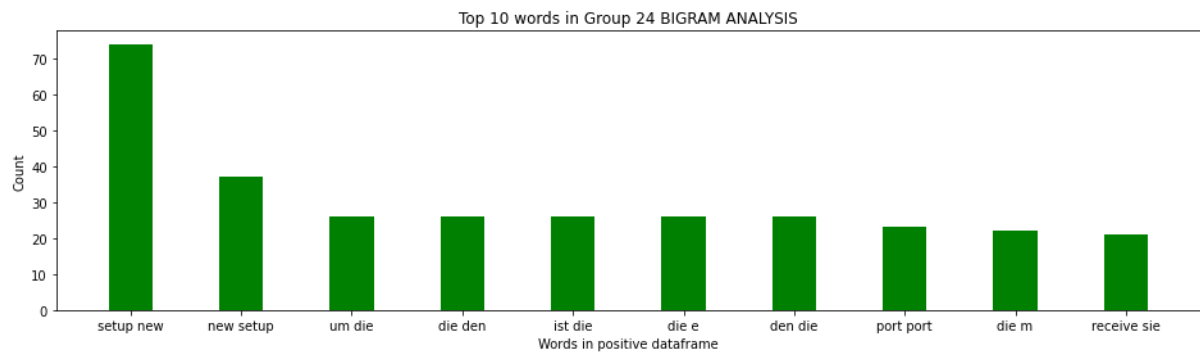
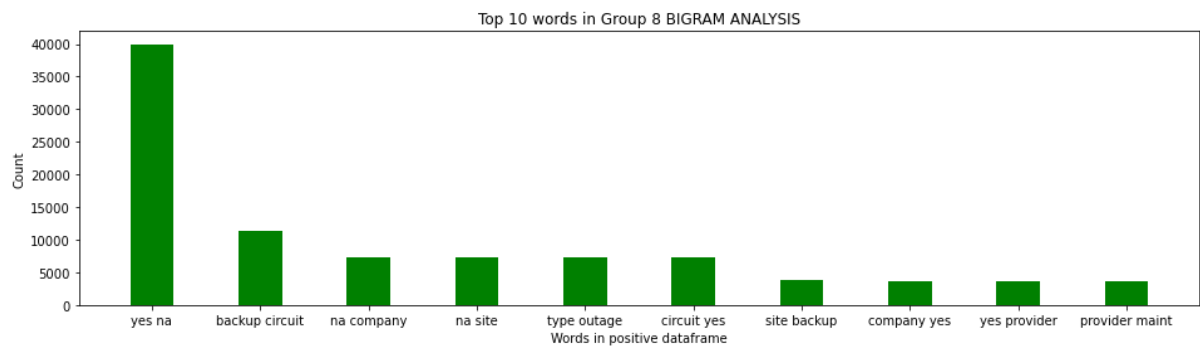
Grp_12



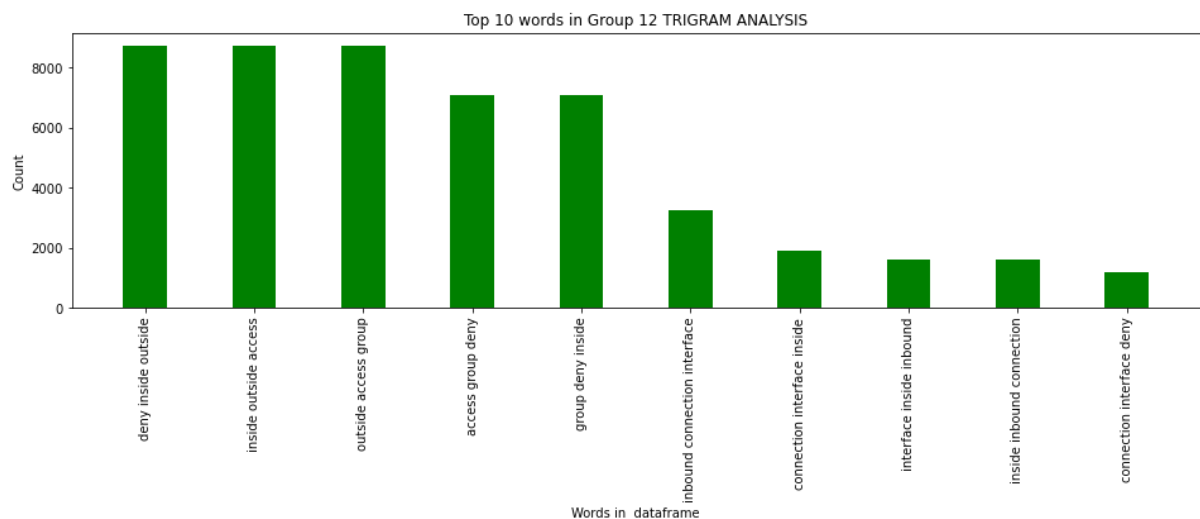
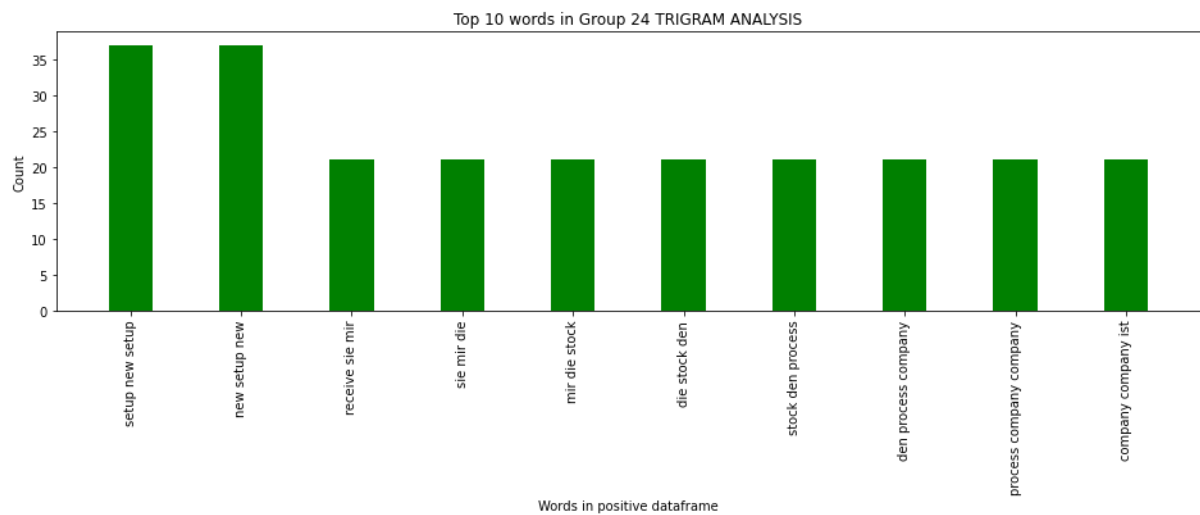
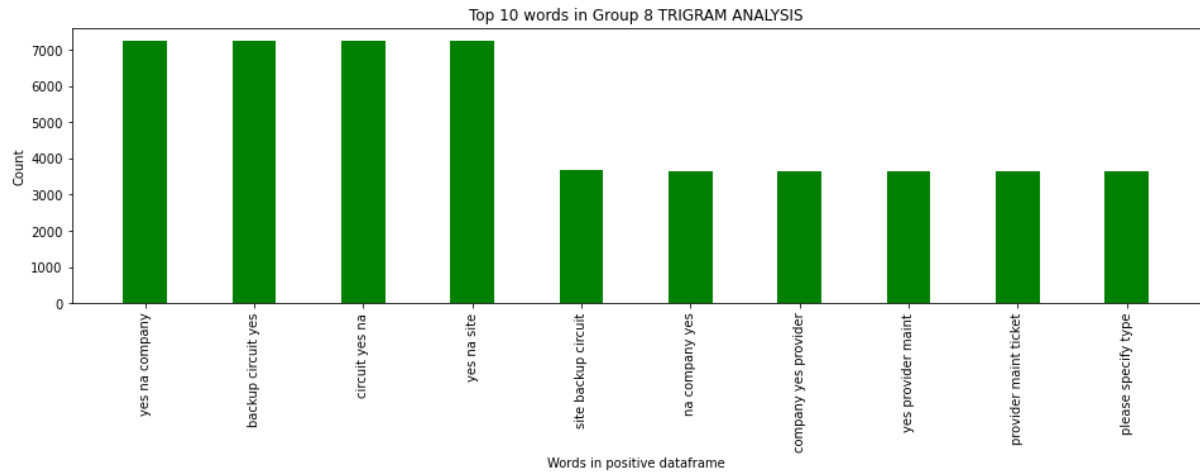
Unigram



Bigram



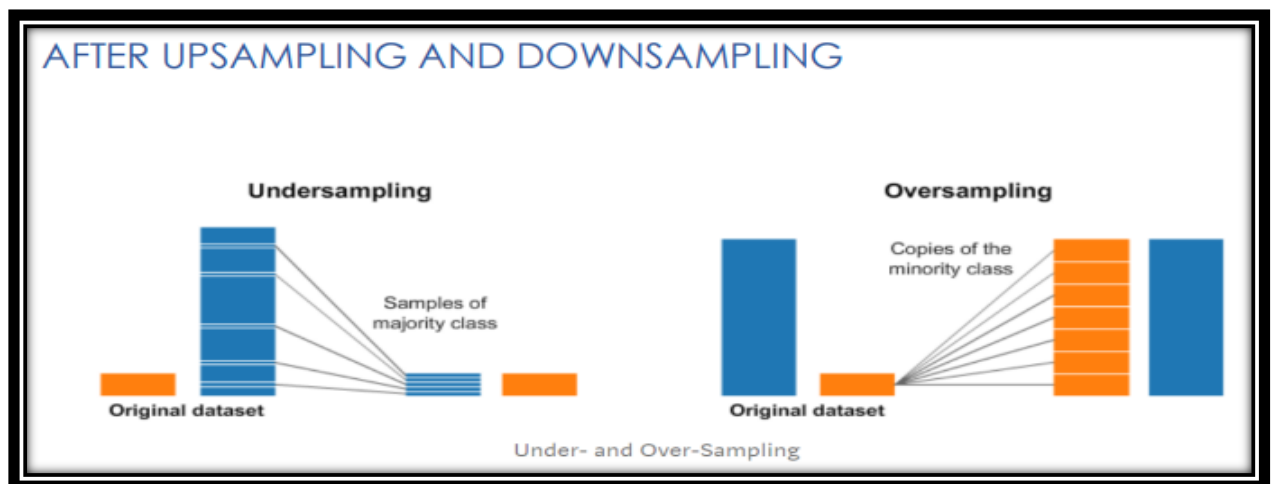
Trigram



Decide Model and Model building

As the target class is completely skewed, various models have been tried with the sampled datasets to compare each performance.

1. Unigram, Bi-gram and Tri-gram models
2. Glove Embedding
3. Count Vectorization
4. TF - IDF Vectorization
5. Text Augmentation
6. Naive Bayes
7. Logistic Regression
8. XGBoost
9. GRU model
10. RNN model
11. Random Forest
12. Bidirectional LSTM



Word Embedding

As all our Machine Learning and Deep learning algorithms are incapable of processing strings or plain text in their raw form, word embedding has been used to convert the texts into numbers. There may be different numerical representations of the same text. It tries to map a word using a dictionary to a vector.

Tokenizing Text -> Representing each word by a number Mapping of original word to number is preserved in word index property of tokenizer Tokenized applies basic processing like changing it to lower case, explicitly setting that as False Lets keep all news to 300, add padding to news with less than 300 words and truncating long ones.

We have experimented with Glove embedding in our models.

GloVe (Global Vectors) Embedding:

GloVe is an unsupervised learning algorithm for obtaining vector representations for



words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Text Augmentation

Data augmentation techniques are used to generate additional, synthetic data using the data you have. We used Synonym Replacement technique to up sample text Randomly choose n words from the sentence that are not stop words. Replace each of these words with one of its synonyms chosen at random.

Count Vectorization

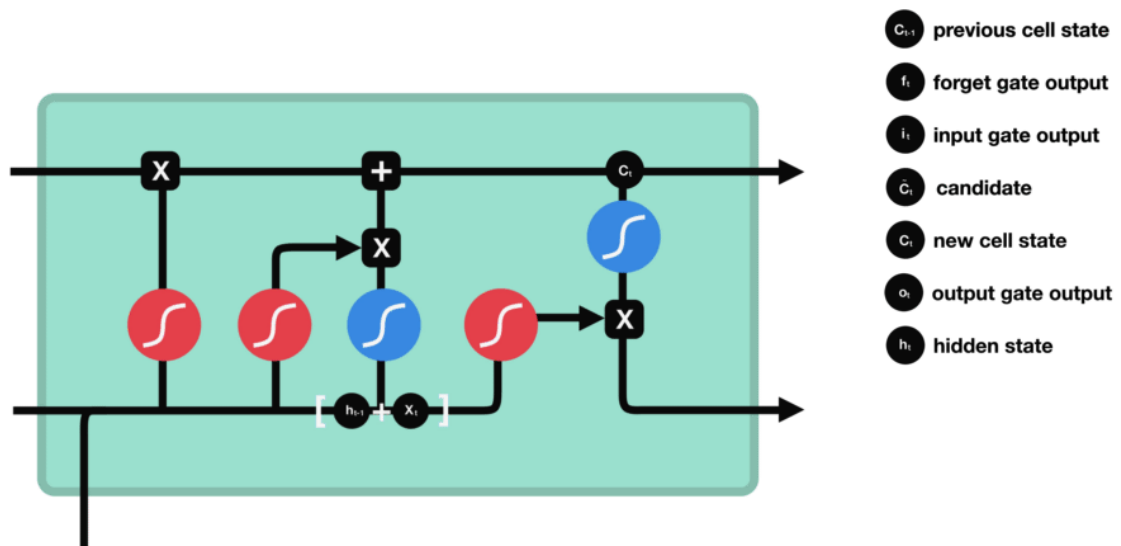
Count Vectorization involves counting the number of occurrences each word appears in a document (i.e., distinct text such as an article, book, even a paragraph!). Python's Sci-kit learn library has a tool called Count Vectorizer to accomplish this.

Tf idf Vectorization

Tf idf Vectorizer - Transforms text to feature vectors that can be used as input to estimator. `vocabulary_` is a dictionary that converts each token (word) to feature index in the matrix, each unique token gets a feature index

Bi-directional LSTM Model

Bidirectional LSTMs are an extension of traditional LSTMs that can improve model performance on classification problems. In problems where all timesteps of the input sequence are available, Bidirectional LSTMs train two instead of one LSTMs on the input sequence. The first on the input sequence as-is and the second on a reversed copy of the input sequence. This can provide additional context to the network and result in faster and even fuller learning on the problem.



Machine Learning Models

Logistic Regression

classification algorithm to predict multi class labels

- Applied logistic regression after applying count vectorization on data

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
	LR_Count_Vector	0.606164	0.670583	0.604288	0.007527	0.606164	0.648491

- Applied logistic regression after applying TF-IDF Vectorization

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	LR_TF-IDF	0.371645	0.402607	0.34853	0.00664	0.371645	0.395404

- Applied logistic regression after applying TF-IDF sequence padding

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
	LR_seq	0.185968	0.366969	0.167797	0.007708	0.185968	0.174948

- Logistic Regression on Original data before Up sampling

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	LR_count_vector_org	0.616467	0.881371	0.578948	0.010855	0.616467	0.581318



Naïve Bayes

- Applied Naïve Bayes after applying count vectorization on data

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
	NB_Count_Vectors	0.340926	0.360524	0.319414	0.0092	0.340926	0.377817

- Applied Naïve Bayes after applying TF-IDF Vectorization

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	NB_TF-IDF	0.294022	0.316722	0.270852	0.01329	0.294022	0.375193

- Applied Naïve Bayes after applying TF-IDF sequence padding

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	NB_seq	0.106019	0.121039	0.079185	0.003835	0.106019	0.195449

- Naïve Bayes on Original data before Up sampling

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
1	NB_Count_Vectors_org	0.485046	0.588457	0.484405	0.011209	0.485046	0.567751



XGBoost

- Applied Xgboost after applying count vectorization on data

Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
NB_Count_Vectors	0.340926	0.360524	0.319414	0.0092	0.340926	0.377817

- Applied Xgboost after applying TF-IDF Vectorization

Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0 NB_TF-IDF	0.294022	0.316722	0.270852	0.01329	0.294022	0.375193

- Applied Xgboost after applying TF-IDF sequence padding

Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0 NB_seq	0.106019	0.121039	0.079185	0.003835	0.106019	0.195449



Deep Learning Models

RNN Architecture

RNNs perform the same task for every element of a sequence, with the output being dependent on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far.

Model: "sequential_60"

Layer (type)	Output Shape	Param #
embedding_55 (Embedding)	(None, None, 300)	1740900
simple_rnn_68 (SimpleRNN)	(None, None, 40)	13640
simple_rnn_69 (SimpleRNN)	(None, None, 40)	3240
simple_rnn_70 (SimpleRNN)	(None, None, 40)	3240
simple_rnn_71 (SimpleRNN)	(None, 40)	3240
dense_51 (Dense)	(None, 12899)	528859
Total params: 2,293,119		
Trainable params: 552,219		
Non-trainable params: 1,740,900		

RNN Performance

Model	test_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
RNN_seq	0.082171	0.012479	0.0	0.082171	0.006752



LSTM Architecture

It is special kind of recurrent neural network that is capable of learning long term dependencies in data. This is achieved because the recurring module of the model has a combination of four layers interacting with each other.

Model: "sequential_48"

Layer (type)	Output Shape	Param #
embedding_43 (Embedding)	(None, None, 300)	1740900
lstm_22 (LSTM)	(None, 32)	42624
dropout_18 (Dropout)	(None, 32)	0
dense_39 (Dense)	(None, 12899)	425667
Total params: 2,209,191		
Trainable params: 468,291		
Non-trainable params: 1,740,900		

LSTM Performance

	Model	test_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	LSTM_c	0.082171	0.012479	0.0	0.082171	0.006752



GRU Architecture

To solve the vanishing gradient problem of a standard RNN, GRU uses, so-called, **update gate and reset gate**. Basically, these are two vectors which decide what information should be passed to the output. The special thing about them is that they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction.

Model: "sequential_59"

Layer (type)	Output Shape	Param #
embedding_54 (Embedding)	(None, None, 100)	580300
gru_9 (GRU)	(None, 32)	12864
dropout_29 (Dropout)	(None, 32)	0
dense_50 (Dense)	(None, 12899)	425667
Total params: 1,018,831		
Trainable params: 1,018,831		
Non-trainable params: 0		

GRU Performance

	Model	test_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	GRU_WE	0.082558	0.013249	0.00077	0.082558	0.028075





ML Model Performance

	Model	test_accuracy_mean	train_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	LR_Count_Vector	0.606164	0.670583	0.604288	0.007527	0.606164	0.648491
0	LR_count_vector_org	0.616467	0.881371	0.578948	0.010855	0.616467	0.581318
0	NB_Count_Vectors_org	0.485046	0.588457	0.484405	0.011209	0.485046	0.567751
0	LR_TF-IDF	0.371645	0.402607	0.34853	0.00664	0.371645	0.395404
0	NB_Count_Vectors	0.340926	0.360524	0.319414	0.0082	0.340926	0.377817
0	NB_TF-IDF	0.294022	0.316722	0.270852	0.01329	0.294022	0.375193
0	LR_seq	0.185968	0.366969	0.167797	0.007708	0.185968	0.174948
0	NB_seq	0.106019	0.121039	0.079185	0.003835	0.106019	0.195449

DL Model Performance

	Model	test_accuracy_mean	test_f1-score_mean	test_f1-score_std	test_recall_score_mean	test_precision_score_mean
0	GRU_WE	0.082558	0.013249	0.00077	0.082558	0.028075
0	RNN_WE	0.082171	0.012479	0.0	0.082171	0.006752
0	RNN_seq	0.082171	0.012479	0.0	0.082171	0.006752
0	LSTM_c	0.082171	0.012479	0.0	0.082171	0.006752
0	CNN_GRU_WE	0.082171	0.012479	0.0	0.082171	0.006752





Comparison to benchmark

From the given problem description, we could see that the current system is able to assign 75% of the tickets correctly. So, our objective here is to build an AI-based classifier model to assign the tickets to right functional groups by analysing the given description and achieve a higher degree of accuracy. From the prediction results we see that the GRU model based on is able to achieve an accuracy of 82.25% which is above than the benchmark.

Implications

Although the GRU model can classify the IT tickets with 82.25% accuracy, to achieve better accuracy in the real world it would be good if additional data is collected for each group which may help reduce the data imbalance and skewness of the data.

Limitations

As part of Data pre-processing, we have considered only the top 20 groups based on the count frequency. However, when applying this model in real world scenario, there could be quite a deviation in the model performance if the data is balanced across all assignment groups.

Closing Reflections

We found the data was present in multiple languages and in various formats such as emails, chat, etc bringing in a lot of variability in the data to be analyzed.

The Business can improve the process of raising tickets via a common unified IT Ticket Service Portal which reduces the above-mentioned variability.

By doing this, the model can perform better which can help businesses to identify the problem area for relevant clusters of topics.

Acknowledgement

During the course of this entire engagement, many people have helped us. We would like to thank all those who have been associated with this project, starting with our Program Manager Mr Pratish Kalyani.

Thanks to our Mentor Mr Sandeep Raghuwanshi for his suggestions and directions which helped us the completion of this project. We would also like to extend our thanks and regards to all the mentors who taught us and explained us the various techniques and concepts so beautifully and the support staff at Great Lakes and Great Learning who have helped us in various things all throughout the program.

Finally, we would like to thank our family and friends who have been very helpful in various stages of project completion.

Regards
Group 10 NLP 1

