Automatic Ticket Assignment (NLP)

OCTNLPGroup1 - AIML2020-2021

Milestone – 1 Submission

Mentor : **Mr. Sahil Mattoo**

Program Coordinator : **Mr. Christien Abraham**

Team Members:

- ✓ Saurabh Mishra
- ✓ Krishnamurthy S
- ✓ Anuradha Kini
- ✓ Kavitha R
- ✓ Preeti Binu

1. Summary of the problem statement, Data and findings

1.1. Problem Statement:

One of the key activities of any IT function is to “Keep the lights on” to ensure there is no impact to the Business operations. IT leverages Incident Management process to achieve the above Objective. An incident is something that is unplanned interruption to an IT service or reduction in the quality of an IT service that affects the Users and the Business. The main goal of Incident Management process is to provide a quick fix / workarounds or solutions that resolves the interruption and restores the service to its full capacity to ensure no business impact. In most of the organizations, incidents are created by various Business and IT Users, End Users/ Vendors if they have access to ticketing systems, and from the integrated monitoring systems and tools. Assigning the incidents to the appropriate person or unit in the support team has critical importance to provide improved user satisfaction while ensuring better allocation of support resources.

The assignment of incidents to appropriate IT groups is still a manual process in many of the IT organizations. Manual assignment of incidents is time consuming and requires human efforts. There may be mistakes due to human errors and resource consumption is carried out ineffectively because of the misaddressing. On the other hand, manual assignment increases the response and resolution times which result in user satisfaction deterioration / poor customer service.

1.2. Data & Findings

The input data provided as an excel sheet. It has the following details:

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

Sample Data:

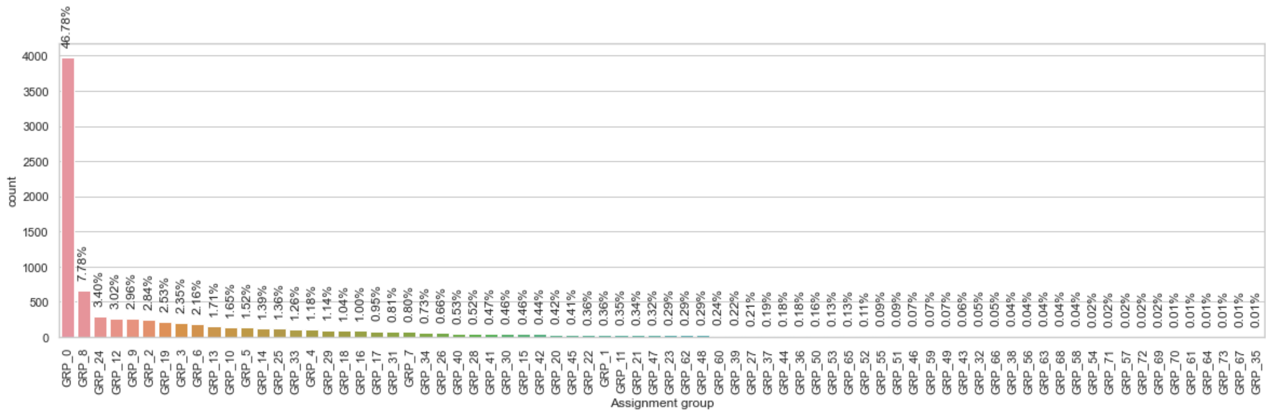
Short description	Description	Caller	Assignment group
wifi disconnecting	\r\n\r\nreceived from: puxiomgy.ndjorwab@gmail...	puxiomgy ndjorwab	GRP_19
desktop notworking	desktop connected to the phone system in serve...	jvhqyamt wodzrcjg	GRP_19
reset ess password	reset ess password	apgukfow soqdkxtb	GRP_0
EU_tool, chargenverwaltung , pdv	die systeme EU_tool, chargenverwaltung, pdv la...	hwxqoijt cotsgwrj	GRP_25
vmax disk failed? on df-6a-d-0 is failed	reporting_tool alert:the monitor is the disk f...	oldrctiu bxurpsyi	GRP_8
in erp's md04 for 6999065 it show a delivery n...	calling from plant plant_35. in erp's md04 for...	dsyzveju ivmprauh	GRP_6
routing fix in germany	please add a static route on the lan switch in...	smpijawb eawkpgqf	GRP_4
in erp's md04 for 6999065 it show a delivery n...	calling from plant plant_35. in erp's md04 for...	dsyzveju ivmprauh	GRP_6
desktop notworking	desktop connected to the phone system in serve...	jvhqyamt wodzrcjg	GRP_19
ç”å¤: email address link to delivery not è½¬...	\r\n\r\nreceived from: jkmeusfq.vjpckzsa@gmail...	jkmeusfq vjpckzsa	GRP_18

Data Findings:

1. The dataset has total of 8500 samples.

2. The Assignment Group column is the target variable and classes among which the incidents will be assigned.

3. High imbalance seen in data with Group - GRP_0 having 46.78% of representation.



4. 73 Groups constitutes only 53%

5. Data has Null values:

Columns	Count of NULL values
Short description	8
Description	1
Assignment group	0

6. Observed certain Short descriptions are same as Description.

7. Password reset seems to be highest re-occuring ticket. A specific caller has a very high frequency of raising tickets.

8. 74 target class found.

Implications:

- Data is highly imbalanced between Group-0 and rest groups.
- Data imbalance will be impacting model performance and it will be biased towards majority classes.

2. Overview of the Final Process

The brief approach for the solution is given below :

1. Solution requires model building based on the Classification model approach to predict the ticket details and assigned to expected group for quicker resolution.

2. Data cleansing and Pre-Processing are important to have a good cleaned input dataset for the model to predict the expected output. Hence the data cleansing and pre-processing steps are given in a detailed manner.

3. Visualization has been given to understand the dataset that feed into the model. This also helps to understand the structure of dataset

4. model creation is defined.

   Based on the conventional Machine learning algorithms Logisitic regression, Random Forest classifier, GradientBoostingClassifier, SVM, KNeighboursClassifier.

   The evaluation approach is given as well.

5. Hyper parameter tunning.

6. The benchmarking of outcome has been captured. The performance of the model is tuned based on the different iterations with different parameters

3. EDA and Pre-Processing

Below are the Pre-Processing steps applied while performing Exploratory Data Analysis on the input data.

1. Removal of rows that contains Null values - Impacted 9 rows.
2. Caller could be an important feature But caller column mainly contains the details of the user who raised the incident and the same caller has raised in different tickets in different groups, hence the column is of no much use in our analysis and can be dropped.
3. Description contains the full information, Hence dropped the short description column.
4. Convert each character in a sentence to lowercase character.
5. Remove HTML Tags.
6. Remove punctuations.
7. Remove stopwords.
8. Remove common words like com, hello.
9. Replace Contractions.
10. Stemming was causing invalid words, hence used a lemmatizer.
11. Localization used for English.
12. Null Values are present in Short Description and Description.
13. Remove extra white spaces.
14. Remove accented characters, special character.
15. Translation to English Language.

16. Remove text in square brackets, remove links, remove punctuation and remove word containing words.

17. Remove caller name from the description.

18. Removal of duplicates rows in description.

	Description	Assignment group	text_len	clean_text	language	clean_text_translated	num_words
15	ticket update on inplant_874743	GRP_0	4	ticket update inplant	en	ticket update inplant	3
35	ticket_no1564677-employment status - new non-e...	GRP_0	5	ticket no employment status new non employee	en	ticket no employment status new non employee	7
40	ticket update - inplant_874615	GRP_0	4	ticket update inplant	sv	ticket update inplant	3
59	received from: monitoring_tool@company.com\r\n...	GRP_8	11	job mm zscr dly merktc failed job scheduler at	en	job mm zscr dly merktc failed job scheduler at	9
60	received from: monitoring_tool@company.com\r\n...	GRP_8	11	job job failed job scheduler at	en	job job failed job scheduler at	6
...	...	...	...	...	...	...	...
8460	received from: monitoring_tool@company.com\r\n...	GRP_9	11	abended job job scheduler job	sl	abended job job scheduler job	5
8462	received from: monitoring_tool@company.com\r\n...	GRP_9	11	abended job job scheduler job	da	abended job job scheduler job	5
8466	received from: monitoring_tool@company.com\r\n...	GRP_8	11	abended job job scheduler bkwin hostname inc	en	abended job job scheduler bkwin hostname inc	7
8486	ticket update on ticket_no0427635	GRP_0	4	ticket update ticket no	sv	ticket update ticket no	4
8489	account locked	GRP_0	2	account locked	en	account locked	2

1882 rows × 7 columns

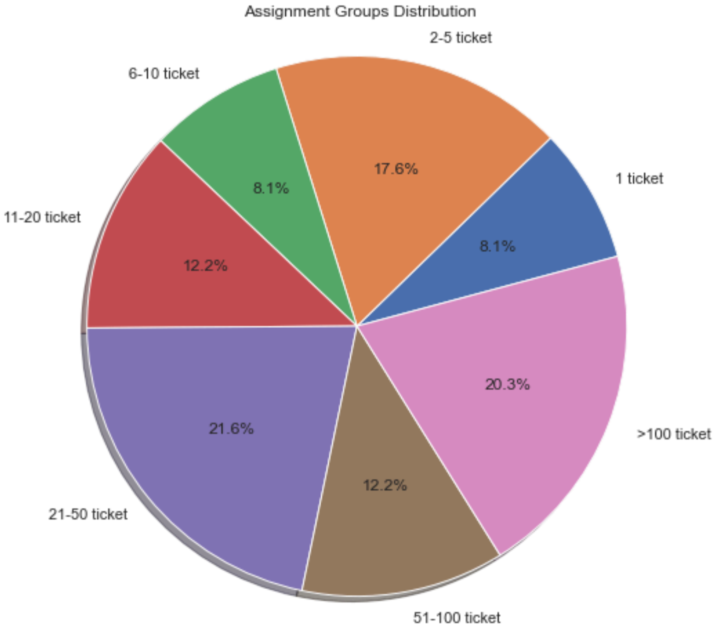
19. Removal of common words across all the groups and retained in the groups where its occurred max number of times like “job scheduler”.

20. Remove the rows that contains less than 2 words.

21. Combining less sample data into new group

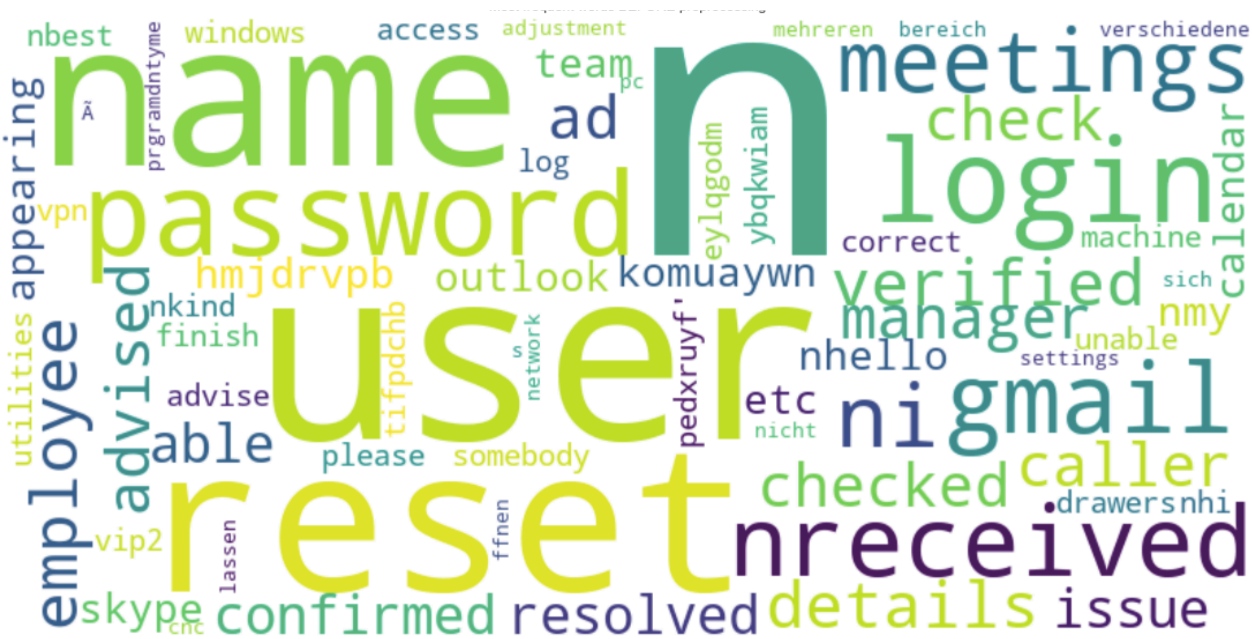
4. Visualization

4.1 Assignment Group vs Ticket distribution:



	Description	Ticket Count
0	1 ticket	6
1	2-5 ticket	13
2	6-10 ticket	6
3	11-20 ticket	9
4	21-50 ticket	16
5	51-100 ticket	9
6	>100 ticket	15

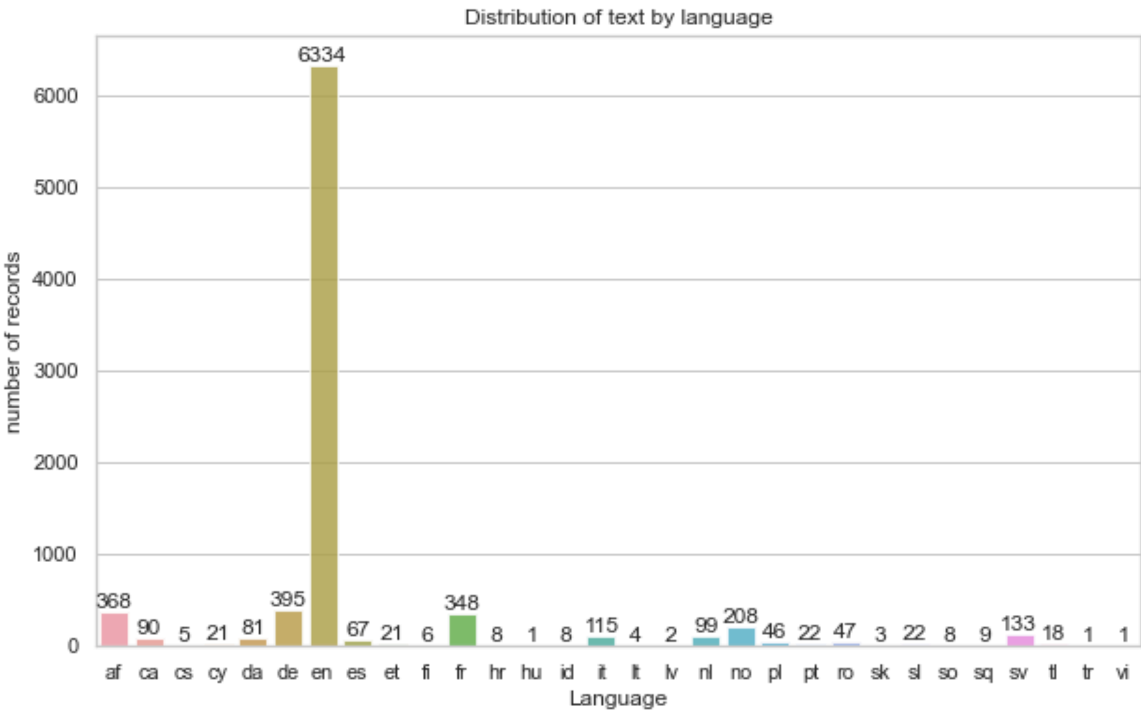
4.2 Most frequent words before preprocessing:



with the help of wordCloud library and stop words builded word cloud on description column.

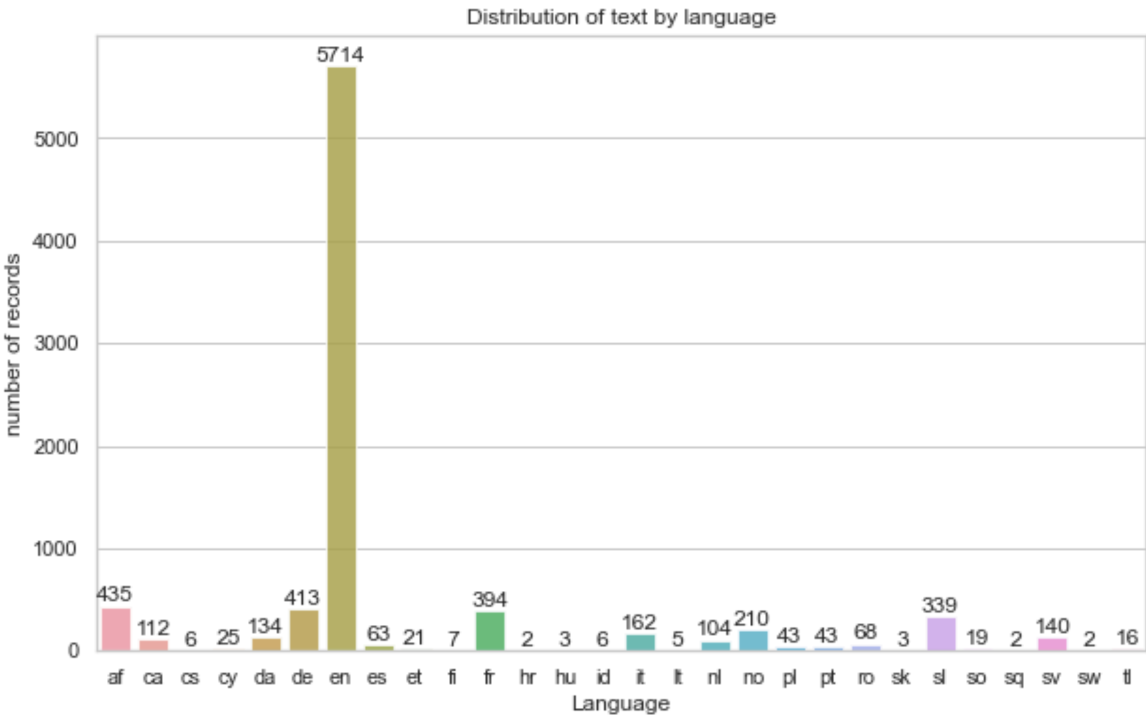
4.3 Distribution of words by language:

we have seen highest ticket raised in english language, we need to convert the other language tickets to english language.



4.4 **Distribution of words by language:**

After the data cleanup we have builded the plot to see the words language



4.4 Most frequent words after preprocessing:

After cleaning the data we have builded the below word cloud.

we can see reset password user name gmail etc are occurring most.

5. Sampling and Feature selection:

- Balance the data to avoid overfitting problem, in group zero taken 500 sample and rest other group by combining them taken 350 samples.

- Label Encoding for Assignment group(target class)

Assignment group		clean_text_translated	Assignment group	Assignment_label
GRP_0	587	outlook skype responding	GRP_0	0
	5818	dear it continue skype audio issue unable hear...	GRP_0	0
	196	ic welcome next available agent shortly ic int...	GRP_0	0
	2577	efyumrls gqjcbufx finance administration manag...	GRP_0	0
	2075	password change ca n t sign skype dell happene...	GRP_0	0

- Vectorize the data using Tf-IDF vector (max feature taken is 20000)

- using the chi-square test measure "weeds-out" the features that are most likely to independent of class and therefore irrelevant for classification.

6. Model building and evaluation

6.1 Model Approach

Solution requires model building based on the Multi Classification model approach to predict the ticket details and assigned to expected group for quicker resolution.

We can approach solution with both Conventional model and using NLP.

In Conventional Model, we are using Logistic regression, Random Forest and KNN models to predict the ticket group

6.2 Model creation

Following Model and accuracy scores are given as per the initial interim stage.

Further Model tuning and performance has been given in the next section

1. Logistic Regression Model

Using TfidfTransformer library in sklearn, bag of words is created to get the vocabulary (ngram 1,2)

Using Vectorizer transformation, features are mapped to training.

Logistic regression model is created and trained with 80-20 train test split.

```
Accuracy: 0.834
F1 score: 0.828
Precision: 0.829
Recall: 0.834
```

2. Random Forest Model :

```
Accuracy: 0.834
F1 score: 0.828
Precision: 0.829
Recall: 0.834
```

3. KNeighborsClassifier :

```
Accuracy: 0.730
F1 score: 0.757
Precision: 0.839
Recall: 0.730
*****************************
```

4. GradientBoostingClassifier :

```
Accuracy: 0.901
F1 score: 0.899
Precision: 0.901
Recall: 0.901
```

5. SVM :

```
Accuracy: 0.898
F1 score: 0.900
Precision: 0.906
Recall: 0.898
```

6. MultinomialNB:

```
Accuracy: 0.806
F1 score: 0.804
Precision: 0.825
Recall: 0.806
****************************************************
```

6.3Models After Hyper Parameter Tuning:

1. Logistic Regression Tunning:

```
Accuracy: 0.902
F1 score: 0.900
Precision: 0.900
Recall: 0.902
****************************************************
```

2. SVM tunning:

```
Accuracy: 0.915
F1 score: 0.916
Precision: 0.918
Recall: 0.915
```

3. Random Forest Classifier tunning:

```
Accuracy: 0.920
F1 score: 0.920
Precision: 0.923
Recall: 0.920
```

4. KNeighbours Classifier Tunning:

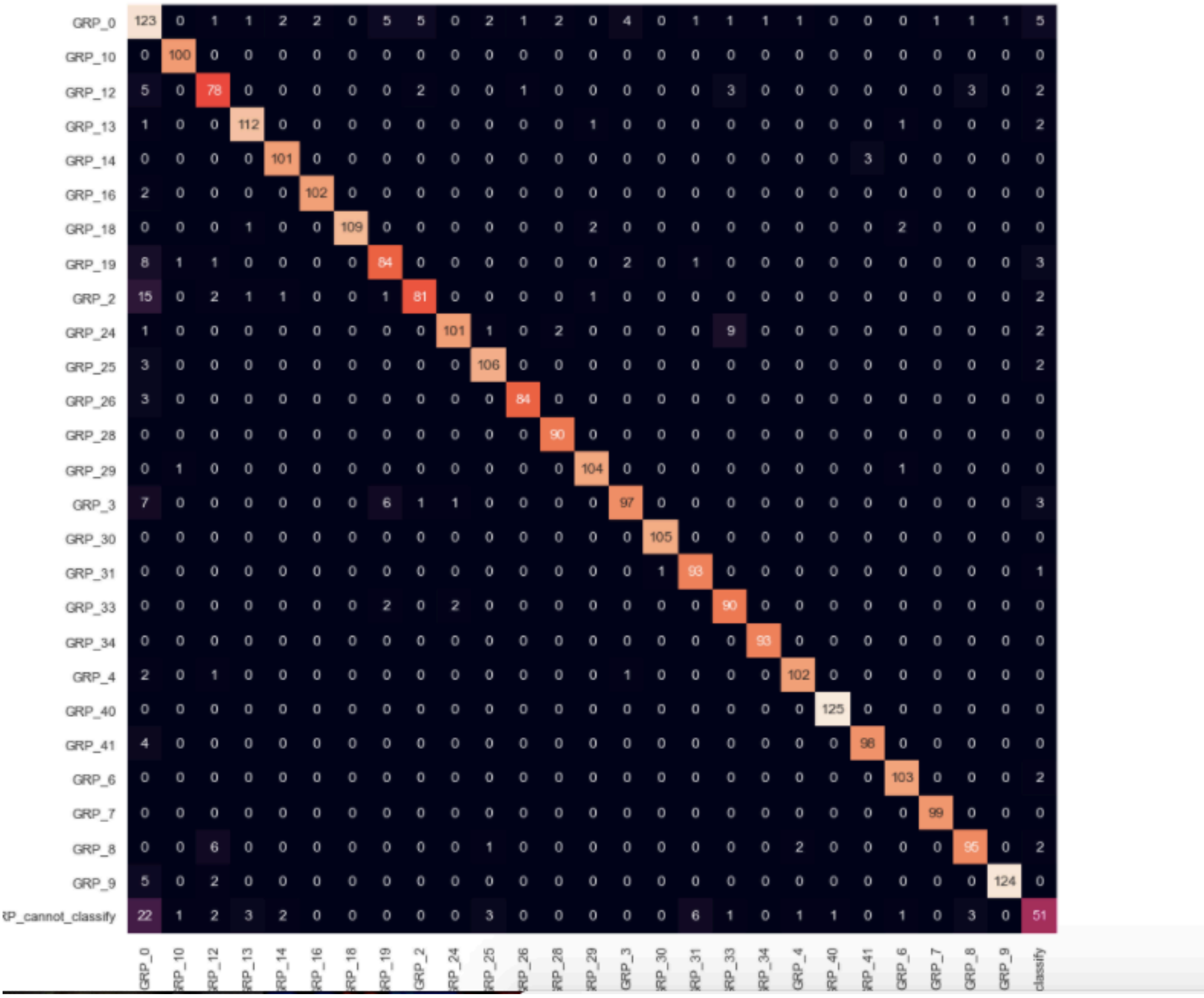
```
Accuracy: 0.889
F1 score: 0.898
Precision: 0.918
Recall: 0.889
*************************************************
```

6.4 Model Summary:

Summary of Model outputs

	Model	Training Score	Testing Score
7	RandomForestClassifier Tunned	0.998958	0.920139
2	Random Forest Classifier	0.998958	0.916319
8	SVC-Tunned	0.995387	0.915278
3	GradientBoostingClassifier	0.989137	0.903819
6	LogisticRegression-Tunned	0.985714	0.902431
4	SVM	0.976488	0.897917
9	KNN Tunned	0.998958	0.889236
1	LogisticRegression	0.905060	0.833681
0	MultinomialNB	0.881250	0.806250
5	KNN	0.871875	0.730208

6.5 Confusion Matrix for RFC Model:

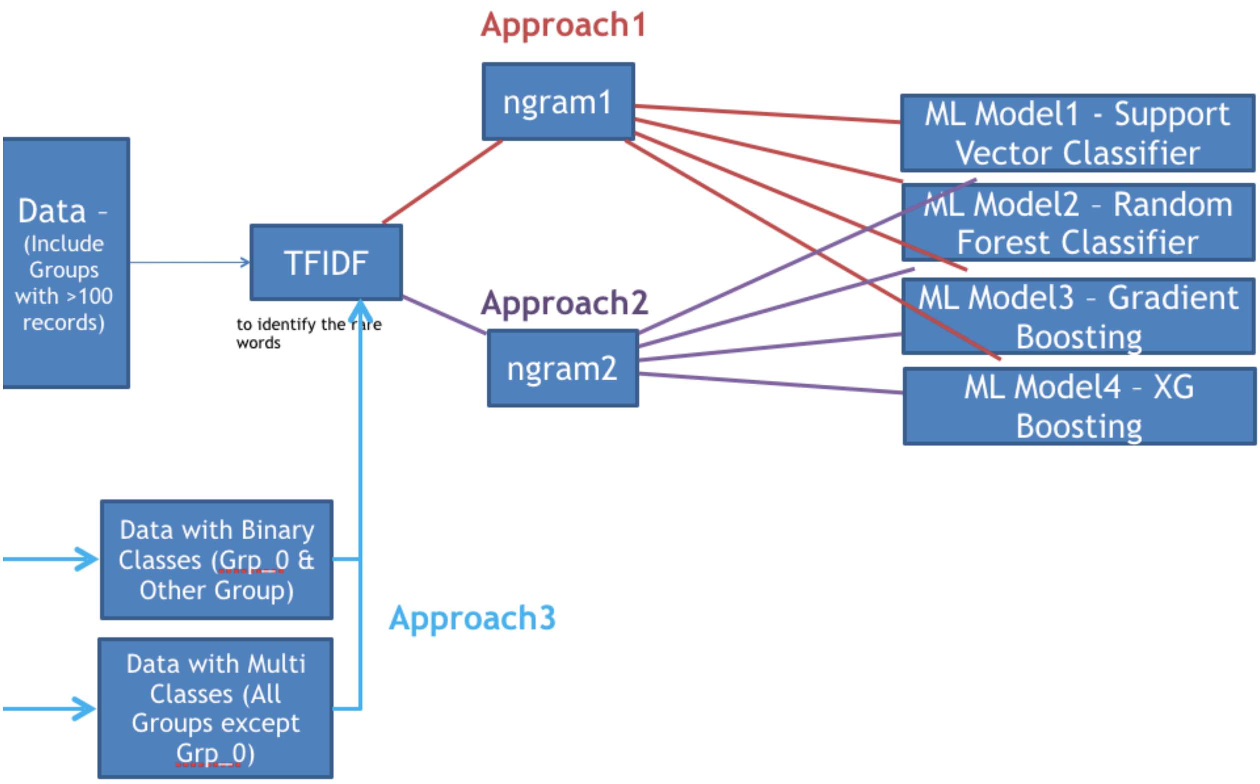


6.6 Predicting Sample:

```
[['cant log in to vpn ']]

predicted group is ['GRP_0']
```

6.7 Alternates ways to build the models:



- We tried to 2 approaches to build the conventional machine learning models as mentioned above:

6.7.1 SVM model using the Ngram(1,1):

```
train accuracy 0.7837837837837838
train f1 score 0.8324411098509058

accuracy 0.7313746065057712
f1 score 0.8031029413012807
              precision    recall  f1-score   support

           0       0.72      0.99      0.83       596
           1       0.57      0.27      0.37        44
           2       1.00      0.19      0.31        27
           3       0.86      0.27      0.41        22
           4       0.00      0.00      0.00        41
           5       0.73      0.18      0.29        45
           6       0.74      0.67      0.71        43
           7       0.00      0.00      0.00        23
           8       1.00      0.03      0.05        39
           9       1.00      0.24      0.38        21
          10       0.93      0.81      0.87        52

    accuracy                           0.73       953
   macro avg       0.69      0.33      0.38       953
weighted avg       0.71      0.73      0.66       953
```

6.7.2. SVM model using ngram (1,2):

```
train accuracy 0.7924429283652584
train f1 score 0.836485496583072

accuracy 0.7303252885624344
f1 score 0.8014744245783231
              precision    recall  f1-score   support

           0       0.72      0.99      0.83       596
           1       0.59      0.30      0.39        44
           2       1.00      0.22      0.36        27
           3       0.86      0.27      0.41        22
           4       0.00      0.00      0.00        41
           5       0.73      0.18      0.29        45
           6       0.74      0.65      0.69        43
           7       0.00      0.00      0.00        23
           8       1.00      0.03      0.05        39
           9       0.80      0.19      0.31        21
          10       0.93      0.79      0.85        52

    accuracy                           0.73       953
   macro avg       0.67      0.33      0.38       953
weighted avg       0.70      0.73      0.66       953
```

6.7.3. SVM Model using the ngram(1,2) using the Smote and tfidf:

```
Train dataset
              precision    recall  f1-score   support

           0       1.00      0.98      0.99      2383
           1       1.00      1.00      1.00      2383
           2       1.00      1.00      1.00      2383
           3       1.00      1.00      1.00      2383
           4       1.00      1.00      1.00      2383
           5       0.99      1.00      0.99      2383
           6       0.99      1.00      0.99      2383
           7       1.00      1.00      1.00      2383
           8       1.00      1.00      1.00      2383
           9       1.00      1.00      1.00      2383
          10       1.00      1.00      1.00      2383

    accuracy                           1.00     26213
   macro avg       1.00      1.00      1.00     26213
weighted avg       1.00      1.00      1.00     26213
```

6.7.4. Random Forest using ngram(1,1) TFIDF:

```
train accuracy 0.8344266596693781
train f1 score 0.8532809656177002

accuracy 0.7250786988457503
f1 score 0.7738773146146588
              precision    recall  f1-score   support

           0       0.74      0.96      0.84       596
           1       0.53      0.36      0.43        44
           2       0.78      0.26      0.39        27
           3       0.67      0.27      0.39        22
           4       0.33      0.15      0.20        41
           5       0.71      0.22      0.34        45
           6       0.65      0.60      0.63        43
           7       0.75      0.13      0.22        23
           8       0.50      0.08      0.13        39
           9       0.83      0.24      0.37        21
          10       0.77      0.69      0.73        52

    accuracy                           0.73       953
   macro avg       0.66      0.36      0.42       953
weighted avg       0.70      0.73      0.68       953
```

6.7.5. Random Forest using ngram(1,2), TFIDF:

```
train accuracy 0.8310154815009184
train f1 score 0.851089505683585

accuracy 0.720881427072403
f1 score 0.7692592645801325
              precision    recall  f1-score   support

           0       0.74      0.95      0.84       596
           1       0.59      0.39      0.47        44
           2       0.64      0.26      0.37        27
           3       0.75      0.27      0.40        22
           4       0.47      0.22      0.30        41
           5       0.77      0.22      0.34        45
           6       0.59      0.53      0.56        43
           7       1.00      0.17      0.30        23
           8       0.50      0.08      0.13        39
           9       0.50      0.10      0.16        21
          10       0.66      0.71      0.69        52

    accuracy                           0.72       953
   macro avg       0.66      0.36      0.41       953
weighted avg       0.70      0.72      0.67       953
```

6.7.6. Random Forest using Ngram(1,2) using Smote:

```
Train dataset
              precision    recall  f1-score   support

           0       0.37      0.91      0.53      2383
           1       0.97      0.87      0.91      2383
           2       1.00      0.93      0.96      2383
           3       1.00      0.93      0.96      2383
           4       0.98      0.66      0.79      2383
           5       0.98      0.78      0.87      2383
           6       1.00      0.89      0.94      2383
           7       0.89      0.94      0.91      2383
           8       0.97      0.56      0.71      2383
           9       0.99      0.82      0.90      2383
          10       1.00      0.96      0.98      2383

    accuracy                           0.84     26213
   macro avg       0.92      0.84      0.86     26213
weighted avg       0.92      0.84      0.86     26213
```

6.7.7. Gradient Boost using the Ngram (1,1), TFIDF:

```
train accuracy 0.8068748360010496
train f1 score 0.8362030827808895

accuracy 0.7334732423924449
f1 score 0.7859858798140098
              precision    recall  f1-score   support

           0       0.73      0.97      0.84       596
           1       0.58      0.34      0.43        44
           2       0.86      0.44      0.59        27
           3       0.67      0.27      0.39        22
           4       1.00      0.05      0.09        41
           5       0.78      0.31      0.44        45
           6       0.61      0.51      0.56        43
           7       1.00      0.17      0.30        23
           8       0.60      0.08      0.14        39
           9       1.00      0.24      0.38        21
          10       0.88      0.67      0.76        52

    accuracy                           0.73       953
   macro avg       0.79      0.37      0.45       953
weighted avg       0.75      0.73      0.68       953
```

6.7.8. Gradient Boost using ngram(1,2) TFIDF:

```
train accuracy 0.8228811335607452
train f1 score 0.8469693273387625

accuracy 0.7334732423924449
f1 score 0.7884378339322305
              precision    recall  f1-score   support

           0       0.74      0.98      0.84       596
           1       0.65      0.39      0.49        44
           2       0.83      0.37      0.51        27
           3       0.71      0.23      0.34        22
           4       0.75      0.07      0.13        41
           5       0.70      0.31      0.43        45
           6       0.60      0.49      0.54        43
           7       0.60      0.13      0.21        23
           8       0.67      0.05      0.10        39
           9       0.80      0.19      0.31        21
          10       0.82      0.71      0.76        52

    accuracy                           0.73       953
   macro avg       0.72      0.36      0.42       953
weighted avg       0.73      0.73      0.68       953
```

6.7.9. Gradient boost using ngram (1,2) with Smote for class Balancing:

```
           0       0.74      0.75      0.74      2383
           1       0.92      0.90      0.91      2383
           2       0.98      0.99      0.99      2383
           3       0.96      0.95      0.95      2383
           4       0.87      0.86      0.87      2383
           5       0.94      0.91      0.92      2383
           6       0.97      0.92      0.94      2383
           7       0.88      0.99      0.93      2383
           8       0.89      0.81      0.85      2383
           9       0.92      0.96      0.94      2383
          10       0.94      0.96      0.95      2383

    accuracy                           0.91     26213
   macro avg       0.91      0.91      0.91     26213
weighted avg       0.91      0.91      0.91     26213
```

6.7.10. XG Boost using ngram(1,1) using TFIDF:

```
train accuracy 0.9312516399895041
train f1 score 0.9340721282724858

accuracy 0.7366211962224554
f1 score 0.7746075001478544
              precision    recall  f1-score   support

           0       0.77      0.95      0.85       596
           1       0.60      0.55      0.57        44
           2       0.87      0.48      0.62        27
           3       0.56      0.23      0.32        22
           4       0.28      0.12      0.17        41
           5       0.69      0.20      0.31        45
           6       0.63      0.63      0.63        43
           7       0.56      0.22      0.31        23
           8       0.42      0.13      0.20        39
           9       0.62      0.38      0.47        21
          10       0.80      0.69      0.74        52

    accuracy                           0.74       953
   macro avg       0.62      0.42      0.47       953
weighted avg       0.71      0.74      0.70       953
```

6.7.11. XG Boost using ngram (1,2) and TFIDF:

```
train accuracy 0.9320388349514563
train f1 score 0.9345283687701192

accuracy 0.7460650577124869
f1 score 0.7830624478439223
```

	precision	recall	f1-score	support
0	0.77	0.95	0.85	596
1	0.58	0.48	0.53	44
2	0.80	0.44	0.57	27
3	0.60	0.27	0.37	22
4	0.42	0.20	0.27	41
5	0.71	0.22	0.34	45
6	0.67	0.70	0.68	43
7	0.75	0.26	0.39	23
8	0.50	0.15	0.24	39
9	0.56	0.24	0.33	21
10	0.83	0.73	0.78	52
accuracy			0.75	953
macro avg	0.65	0.42	0.49	953
weighted avg	0.72	0.75	0.71	953

6.7.12. XG Boost using ngram (1,2) with smote:

Train dataset

	precision	recall	f1-score	support
0	0.94	0.98	0.96	2383
1	0.99	0.99	0.99	2383
2	1.00	1.00	1.00	2383
3	1.00	0.99	1.00	2383
4	1.00	0.99	0.99	2383
5	1.00	0.99	0.99	2383
6	0.99	0.99	0.99	2383
7	1.00	1.00	1.00	2383
8	1.00	0.99	0.99	2383
9	1.00	0.99	1.00	2383
10	1.00	0.99	1.00	2383
accuracy			0.99	26213
macro avg	0.99	0.99	0.99	26213
weighted avg	0.99	0.99	0.99	26213

6.7.13 Split the dataset with >1000 records into two parts

1. Binary dataset contains 2 classes - group_0 and otherGroups.

2. Mulitclass dataset contains multi classes except groups 0

Train the two model on train dataset using Ngram (1,1) , (1,2) and Smote:

Model with Ngram (1,1):

```
train accuracy 0.9149829441091577
train f1 score 0.9165810164516414

accuracy 0.8310598111227702
f1 score 0.8351518517902117
              precision    recall  f1-score   support

           0       0.83      0.92      0.87       596
           1       0.84      0.68      0.75       357

    accuracy                           0.83       953
   macro avg       0.83      0.80      0.81       953
weighted avg       0.83      0.83      0.83       953

[[548  48]
 [113 244]]
```

Model with Ngram (1,2) :

```
train accuracy 0.9170821306743637
train f1 score 0.9186407444404898

accuracy 0.8310598111227702
f1 score 0.8351518517902117
              precision    recall  f1-score   support

           0       0.83      0.92      0.87       596
           1       0.84      0.68      0.75       357

    accuracy                           0.83       953
   macro avg       0.83      0.80      0.81       953
weighted avg       0.83      0.83      0.83       953

[[548  48]
 [113 244]]
```

6.7.14. Binary with Smote and Ngram(1,2):

```
Train dataset
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      2383
           1       0.98      0.97      0.98      2383

    accuracy                           0.98      4766
   macro avg       0.98      0.98      0.98      4766
weighted avg       0.98      0.98      0.98      4766

[[2333   50]
 [  63 2320]]

Test Dataset
              precision    recall  f1-score   support

           0       0.60      0.69      0.64       596
           1       0.63      0.53      0.58       596

    accuracy                           0.61      1192
   macro avg       0.62      0.61      0.61      1192
weighted avg       0.62      0.61      0.61      1192

[[412 184]
 [278 318]]
```

6.7.15 Multiclassifier - All Group except group 0 (Balance using Smote) - ngram (1,1):

Train dataset

	precision	recall	f1-score	support
0	0.94	0.94	0.94	207
1	1.00	1.00	1.00	207
2	0.98	0.96	0.97	207
3	0.93	0.97	0.95	207
4	0.93	0.98	0.96	207
5	0.99	0.92	0.95	207
6	0.98	0.99	0.99	207
7	0.95	0.92	0.94	207
8	0.95	1.00	0.97	207
9	0.96	0.93	0.95	207
accuracy			0.96	2070
macro avg	0.96	0.96	0.96	2070
weighted avg	0.96	0.96	0.96	2070

```
[[195   0   1   0   3   0   1   1   2   4]
 [  0 206   0   0   0   0   1   0   0   0]
 [  5   0 198   0   2   0   0   0   0   2]
 [  0   0   0 200   0   1   0   5   1   0]
 [  0   1   0   2 203   0   0   1   0   0]
 [  2   0   0   3   1 191   1   2   7   0]
 [  0   0   0   0   2   0 205   0   0   0]
 [  0   0   1   6   6   0   1 191   1   1]
 [  0   0   0   1   0   0   0   0 206   0]
 [  6   0   3   3   1   1   0   0   0 193]]
```

Test Dataset

	precision	recall	f1-score	support
0	0.05	0.10	0.06	52
1	0.00	0.00	0.00	52
2	0.04	0.02	0.03	52
3	0.20	0.46	0.28	52
4	0.10	0.13	0.11	52
5	0.18	0.06	0.09	52
6	0.00	0.00	0.00	52
7	0.06	0.13	0.08	52
8	0.00	0.00	0.00	52
9	0.11	0.04	0.06	52
accuracy			0.09	520
macro avg	0.07	0.09	0.07	520
weighted avg	0.07	0.09	0.07	520

```
[[ 5  0  4 14 11  2  2 11  2  1]
 [ 4  0  0  7  5  1  0 32  3  0]
 [17  0  1  6  9  0  4 11  4  0]
 [ 7  3  2 24  4  1  1  5  1  4]
 [11  2  2  6  7  1  0 19  4  0]
 [ 8  0  4 19  5  3  0  9  2  2]
 [14  0  4  4  5  3  0 20  2  0]
 [11  1  4 17  6  3  1  7  0  2]
 [16  0  3 16  3  2  0  4  0  8]
 [15  0  1  8 16  1  1  8  0  2]]
```

6.7.16. Multi Classifier - All groups except group_0 (Balance using the Smote) - ngram (1,2)

Train dataset

	precision	recall	f1-score	support
0	0.93	0.93	0.93	207
1	1.00	0.99	0.99	207
2	0.96	0.95	0.96	207
3	0.93	0.93	0.93	207
4	0.92	0.98	0.95	207
5	0.97	0.90	0.94	207
6	0.98	0.94	0.96	207
7	0.94	0.90	0.92	207
8	0.85	0.98	0.91	207
9	0.96	0.93	0.94	207
accuracy			0.94	2070
macro avg	0.94	0.94	0.94	2070
weighted avg	0.94	0.94	0.94	2070

```
[[192   0   1   0   3   1   0   0   5   5]
 [  0 205   1   0   0   0   1   0   0   0]
 [  6   0 197   0   2   0   0   0   0   2]
 [  0   0   1 192   2   1   0   8   2   1]
 [  1   1   0   1 202   0   0   2   0   0]
 [  0   0   1   1   1 187   1   1  15   0]
 [  1   0   0   0   1   0 194   0  11   0]
 [  0   0   1   9   7   0   2 186   1   1]
 [  0   0   0   0   0   3   0   1 203   0]
 [  6   0   3   3   1   1   0   0   1 192]]
```

Test Dataset

	precision	recall	f1-score	support
0	0.11	0.20	0.15	44
1	0.14	0.04	0.06	27
2	0.00	0.00	0.00	22
3	0.08	0.12	0.09	41
4	0.21	0.22	0.22	45
5	0.20	0.05	0.08	43
6	0.00	0.00	0.00	23
7	0.15	0.23	0.18	39
8	0.18	0.29	0.22	21
9	0.18	0.04	0.06	52
accuracy			0.12	357
macro avg	0.13	0.12	0.11	357
weighted avg	0.14	0.12	0.11	357

```
[[ 9  4  4  8 10  2  0  4  1  2]
 [ 8  1  1  5  4  0  0  5  3  0]
 [ 5  0  0  5  3  1  1  6  1  0]
 [10  0  2  5  3  1  2 12  3  3]
 [11  0  1  9 10  2  2  7  2  1]
 [12  2  5  9  4  2  0  3  5  1]
 [ 7  0  0  2  5  0  0  5  4  0]
 [ 7  0  0  8  2  1  2  9  8  2]
 [ 2  0  1  7  2  1  0  2  6  0]
 [ 9  0 20  8  4  0  1  8  0  2]]
```

6.7.17. Model Summary:

Classifier	train_accuracy	train_f1_score
SVC_ngram11	0.783784	0.832441
SVC_ngram12	0.792443	0.836485
RandomF_ngram11	0.834427	0.853281
RandomF_ngram12	0.831015	0.85109
GradientBoost_ngram11	0.806875	0.836203
GradientBoost_ngram12	0.822881	0.846969
XGBoost_ngram11	0.931252	0.934072
XGBoost_ngram12	0.932039	0.934528
Binary_ngram11	0.91708	0.91864
Binary_ngram12	0.98	0.98
Multi_ngram11	0.96	0.96

6.7.18. Model prediction using AutoML:

```
received from: monitoring_tool@company.com job Job_1854 failed in job_scheduler at: 10/31/2016 01:36:00
GRP_OTHERS
GRP_OTHERS
GRP_0
GRP_8
GRP_8
GRP_0
GRP_8
GRP_8
GRP_0
GRP_8
GRP_8
GRP_0
GRP_8
GRP_8
GRP_0
GRP_2
GRP_12
```

- Auto ML model shows that the models trained on SMOTE are not predicting well because they were overfitting.

6.7.19 . Model prediction using the above builded models:

```
received from: monitoring_tool@company.com job Job_1854 failed in job_scheduler at: 10/31/2016 01:36:00
GRP_OTHERS
GRP_OTHERS
GRP_8
GRP_8
GRP_8
GRP_8
GRP_8
GRP_8
GRP_8
GRP_8
```

as we can see clearly builded models are able to find the right group 8 for assigning the ticket.

7. Closing Reflections:

- We learnt about the real time problem solving using Machine Learning
- Made us to brainstorm on various machine learning and NLP Methodology
- Helped us to learn from the each other in the group and also from mentor on different approaches to solve the nlp problem

8. Business insight.

Due to Manual assignment of tickets to the group causes delay in resolution of tickets. Manual assignment increases the response and resolution times which result in user satisfaction deterioration and poor customer service.

Applying NLP to automatically classify tickets and assign to the right owner in a timely manner to save effort, increase user satisfaction and improves handling of support ticketing system.

9. Future Improvements

- Model is not ready for real time deployment.
- Deployment steps need to fixed.
- To Enhance model performance, trying out new models like Neural Network.
- Transfer learning to use the prebuilt models.
- Word overlaps between groups should be reduced to reduce the misclassification

10. Final Note

Thanks to Great Learning team for helping us to learn AIML and to do this Capstone Project

Thanks to **Sahil** who has helped in many ways to complete this course

Many thanks to our Mentor **Sahil**, his experience in the field of AIML has guided us in learning throughout this course. The team appreciates his patience. His practical knowledge gives us a lot of insights to tackle the issues.

11. References:

- https://www.tensorflow.org/
- https://towardsdatascience.com/
- https://machinelearningmastery.com/
- https://www.greatlearning.in/
- https://www.kaggle.com/

12. Libraries Used:

- pandas
- matplotlib
- seaborn
- numpy
- nltk
- sklearn
- wordcloud
- BeautifulSoup
- langdetect
- googletrans
- Unicodedata