CAPSTONE PROJECT

AUTOMATIC TICKET ASSIGNMENT

BATCH: CAPSTONE 1 NLP A

YEAR: 2019-2020

PARTICIPANTS:

SANTHANA LAKSHMI SELVARAJ

ASHWANI KUMAR

VIGNESH ARASAN

TUSHAR SHAH

Contents

1. Summary of Problem Statement, Data and Findings

Problem Statement

As part of our capstone project, we have decided to address a real-life problem of IT ticket assignment.

For any IT function to ensure smooth business operations, there should be minimal to zero impact / disruptions to users or services. However, any unplanned interruption in service affects end users and overall business. In real life, it has been observed that there are number of tickets gets created and assigned to the group looking at some "words" in the problem statement and may end up with a wrong IT group to tackle the problem. In such cases, it is evident that ticket(s) go through multiple rounds of back and forth and ultimately creates dissatisfaction among users and increases cost of IT group handling ticket.

The manual assignment of tickets might have below disadvantages:

- Delay in assignment of ticket to correct group
- Misinterpretation of issue and hence wrong assignment of ticket
- Delay in resolving an issue with one ticket will start impacting other tickets in queue

Objective:

To avoid this problem of ticket getting assigned to a "wrong" group due to wrong interpretation or any delay, we have undertaken a project for Automated Ticket Assignment, wherein we plan create a model which would apply Natural Language Processing algorithms on the available datapoints from end user and would try to assign the ticket to the correct group.

From the given problem description, we could see that the existing 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 analyzing the given description with an accuracy of at least 85%.

Available Data Points

a. Total of 8500 records with 4 columns namely “Short Description”, Description”, “Caller” and “Assignment Group”
    i. Short Description – User problem is described in brief
    ii. Description – User problem is described at length, includes user mail id as well
    iii. Caller – End user id who has raised support ticket
    iv. Assignment Group – Group to which ticket has been assigned
b. Caller has garbled data and may not be useful in any prediction
c. Description contains mail id of the user, greetings words, symbols, special characters etc. which are again may not be of any use in prediction
d. Few words are combined or statements in description field are not formed correctly
e. Column Description:

```
RangeIndex: 8500 entries, 0 to 8499
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Short description  8492 non-null   object
 1   Description        8499 non-null   object
 2   Caller             8500 non-null   object
 3   Assignment group   8500 non-null   object
dtypes: object(4)
```

i. `memory usage: 265.8+ KB`

Data Findings

Following are the initial findings on the available data.

a. Total of 8500 rows with 4 columns
b. Target variable – Assignment Group has unique 74 values – GRP_0, GRP_1…. GRP_73
c. Total of 9 rows with NaN values wherein 8 are part of short description and 1 in description field

	Short description	Description	Caller	Assignment group
2604	NaN	\r\n\r\nreceived from: ohdrnswl.rezuibdt@gmail...	ohdrnswl rezuibdt	GRP_34
3383	NaN	\r\n-connected to the user system using teamvi...	qftpazns fxpnytmk	GRP_0
3906	NaN	-user unable tologin to vpn.\r\n-connected to...	awpcmsey ctdiuqwe	GRP_0
3910	NaN	-user unable tologin to vpn.\r\n-connected to...	rhwsmefo tvphyura	GRP_0
3915	NaN	-user unable tologin to vpn.\r\n-connected to...	hxripljo efzounig	GRP_0
3921	NaN	-user unable tologin to vpn.\r\n-connected to...	cziadygo veiosxby	GRP_0
3924	NaN	name:wvqgbdhm fwchqjor\nlanguage:\nbrowser:mic...	wvqgbdhm fwchqjor	GRP_0
4341	NaN	\r\n\r\nreceived from: eqmuniov.ehxkcbgj@gmail...	eqmuniov ehxkcbgj	GRP_0
4395	i am locked out of skype	NaN	viyglzfo ajtfzpkb	GRP_0

d. Total of 83 rows with duplicate values

```
duplicateRowsDF = atadf[atadf.duplicated()]
print(duplicateRowsDF)
```

```
                                     Short description  ... Assignment group
51                            call for ecwtrjnq jpecxuty  ...            GRP_0
229                           call for ecwtrjnq jpecxuty  ...            GRP_0
493                       ticket update on inplant_872730  ...            GRP_0
512                                    blank call //gso  ...            GRP_0
667   job bkbackup_tool_powder_prod_full failed in j...  ...            GRP_8
...                                                  ...  ...              ...
7836             probleme mit erpgui \tmqfjard qzhgdoua  ...           GRP_24
8051               issue on pricing in distributor_tool  ...           GRP_21
8093  reset passwords for prgthyuulla ramdntythanjes...  ...           GRP_17
8347                            blank call // loud noise  ...            GRP_0
8405                            unable to launch outlook  ...            GRP_0

[83 rows x 4 columns]
```

e. It has been observed that there are 4148 (more than 50%) rows for which short description field values are part of description fields.

f. Datapoints are skewed towards Grp_0 wherein maximum trouble tickets are assigned to – 3976 total, accounts for around 50% of datapoints. Whereas, there are some groups which have only 1 ticket assigned, and this will need to be looked at during modeling as we feel that this bias/skewness will impact on model accuracy, until we have more datapoints.

2. Overview of the final process

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

Data Pre-Processing:

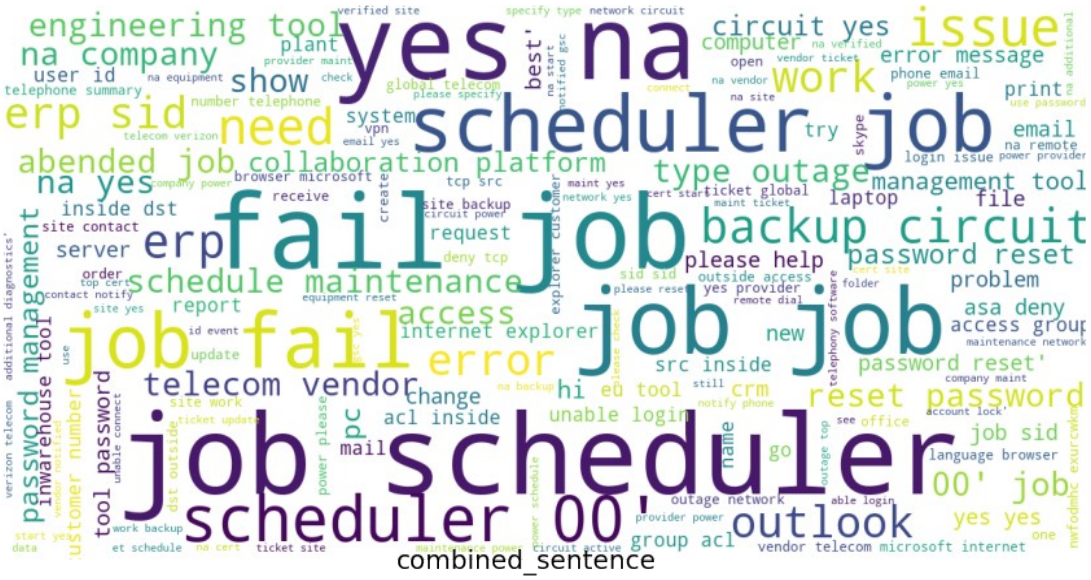
Following pre-processing steps were performed before applying any modeling.

a. Drop Caller field as it contains garbled values and might not be useful at all
b. Null Values – Replace NaN values in Short Description field with corresponding Description field and vice versa
c. Combining Short Description and Description field into a new field "combined_sentence"
d. Translation of German language words to English
e. Tokenization
f. Removal of stop words
g. Lemmatization
h. Cleaning of short description and description fields to remove special characters, greeting words, email addresses etc.
i. Create word cloud
j. Understand word frequency

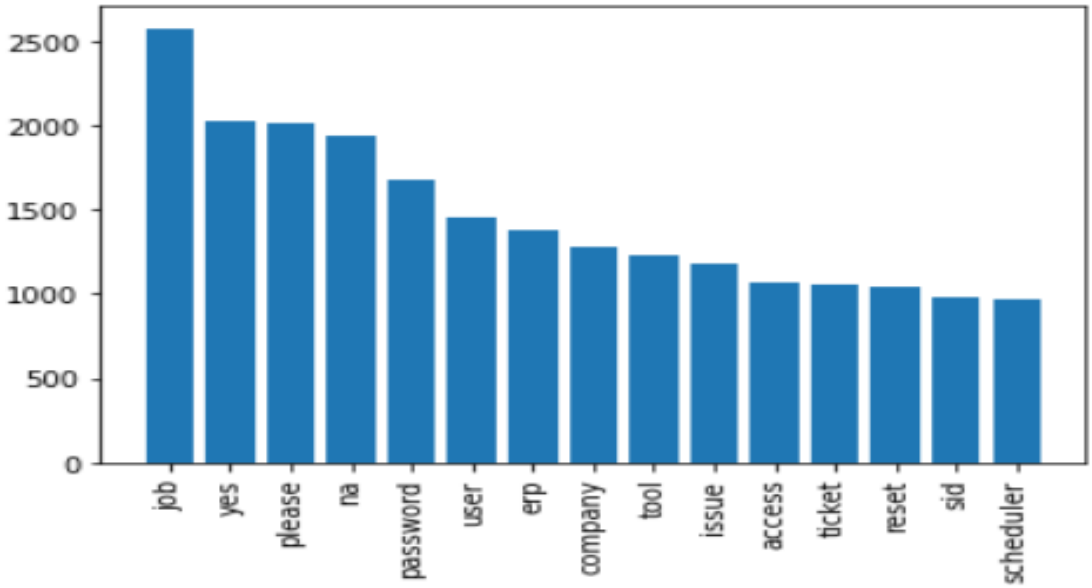
Salient features of the available data:

Following are some of the salient features of the available datapoints.

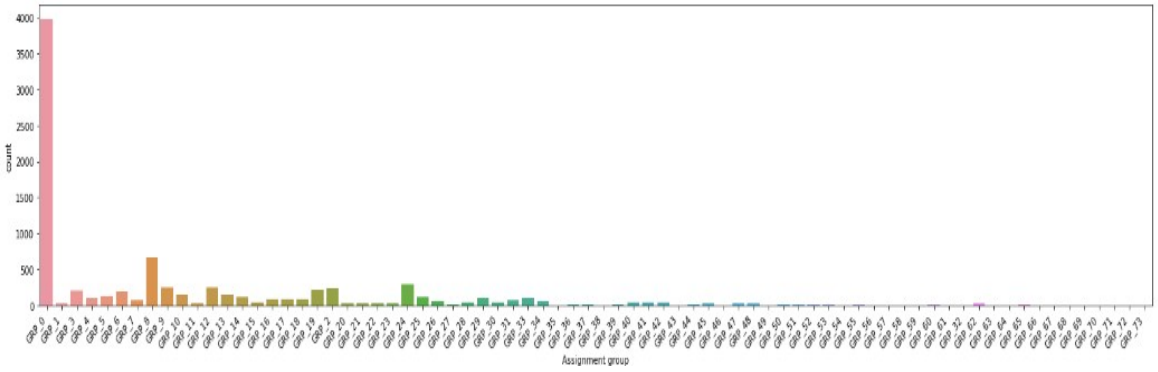
a. Some of the words like Job, Scheduler, Password, user, Fail, Yes, outlook, backup, erp etc. are higher in frequency. These words may help us decide to classify the problem into various groups.



b. Inline to word cloud, we have observed that highest frequency word is Job followed by yes, please, na, password, user, erp etc. These words primarily describe the issues that are occurring the most, as highlighted under description field.



c. Grp_0 is having maximum number of datapoints whereas rest of the groups have comparatively minimal number of datapoints. This gives us an indication that we might have to augment the datapoints under other groups to increase classification accuracy.



Algorithms used for modeling

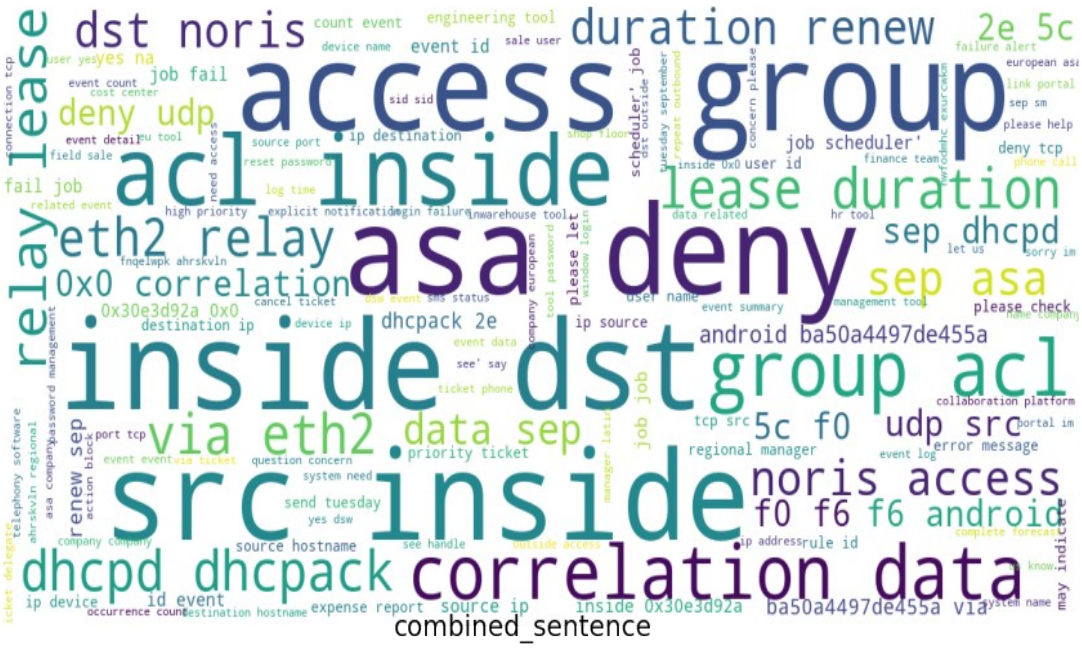
Following algorithms have been tried as part of pre-modelling and final modelling techniques.

a. Tf-IDF vector with Multinomial Naive Bayes
b. Glove Embedding with Bernoulli Naive Bayes
c. Glove embedding with simple LSTM model
d. Glove Embedding with Bidirectional LSTM

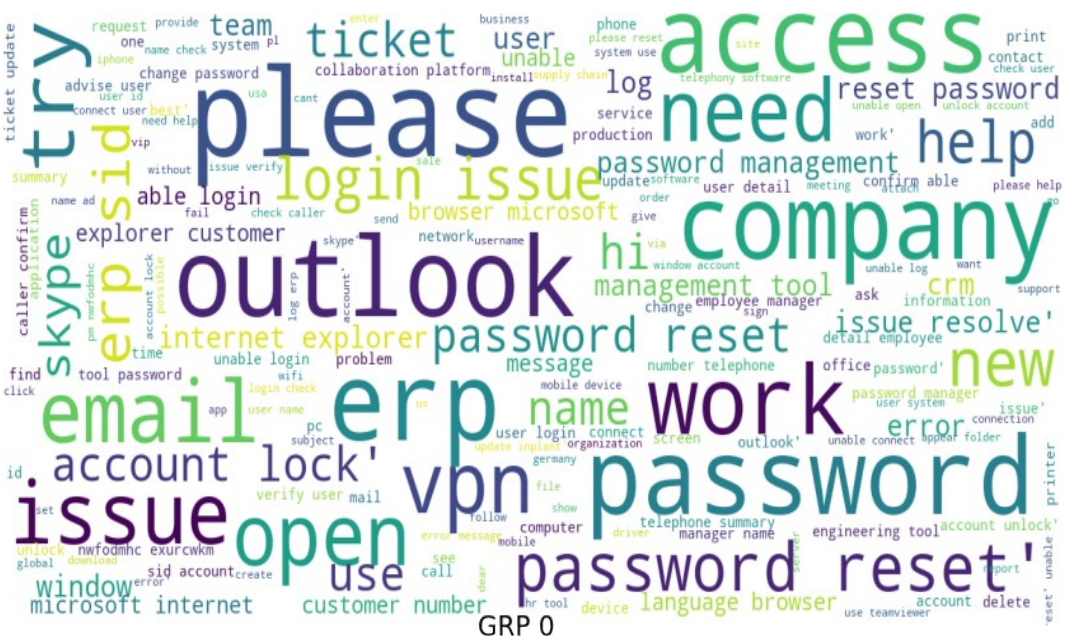
3. Visualization

We have used following visualization on resampled data to really understand high frequency words and association between words in a given group.

a. Word Cloud with Full Dataset

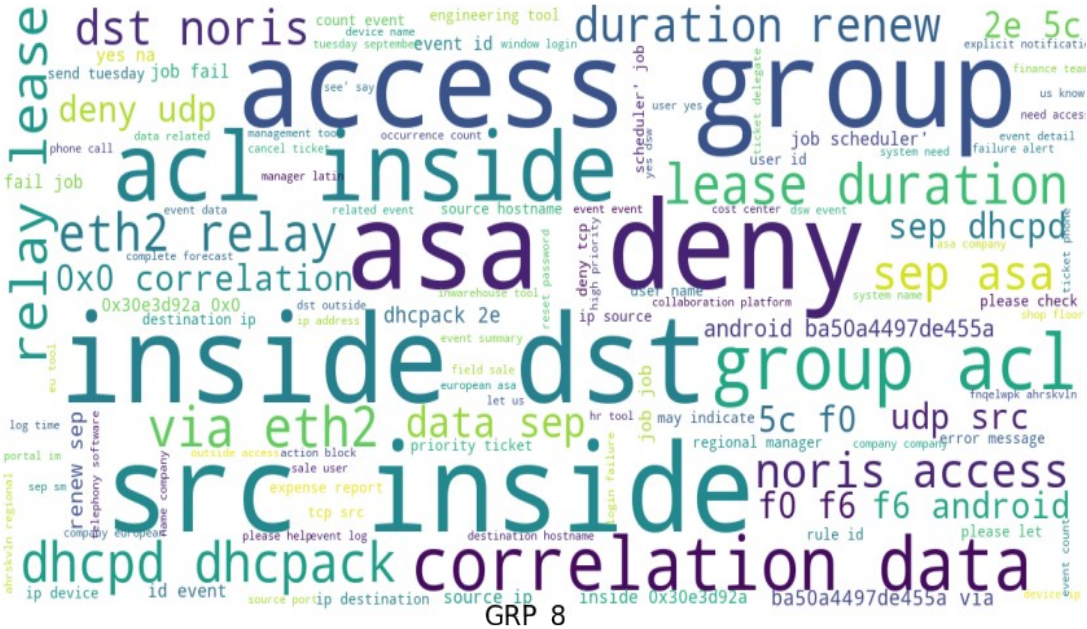


b. Word Cloud for Grp_0

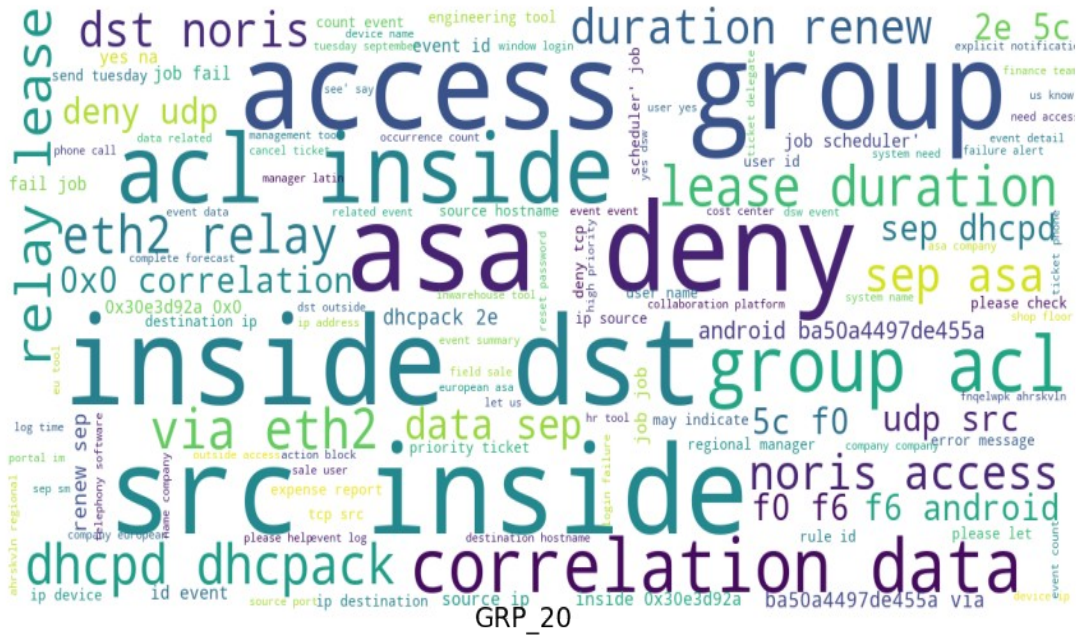


c. Word Cloud for Non Grp_0

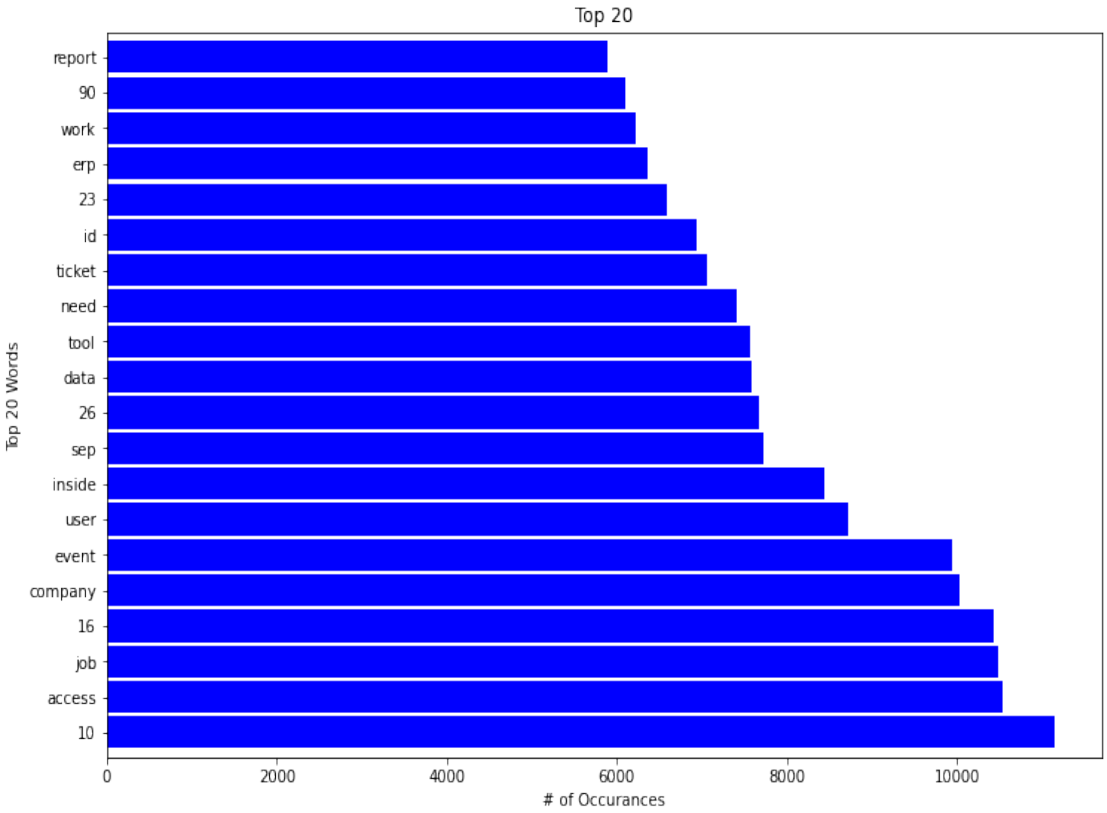
d. Word Cloud for Non Grp_8



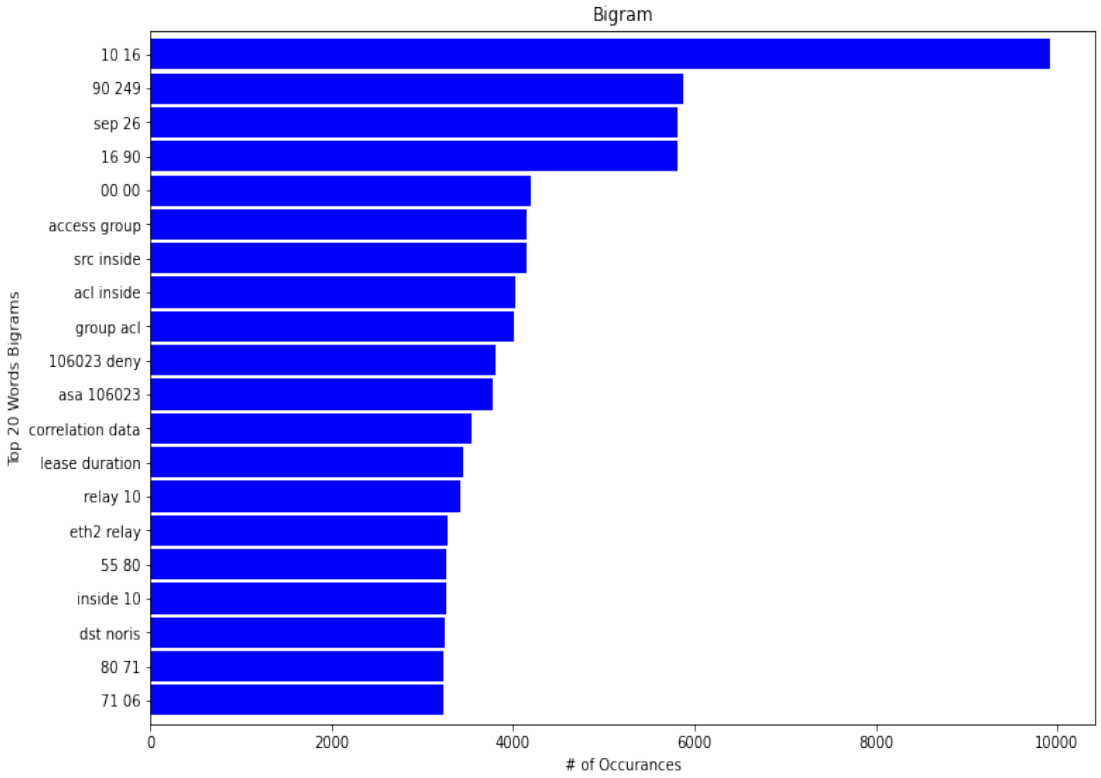
e. Word Cloud for Non Grp_20



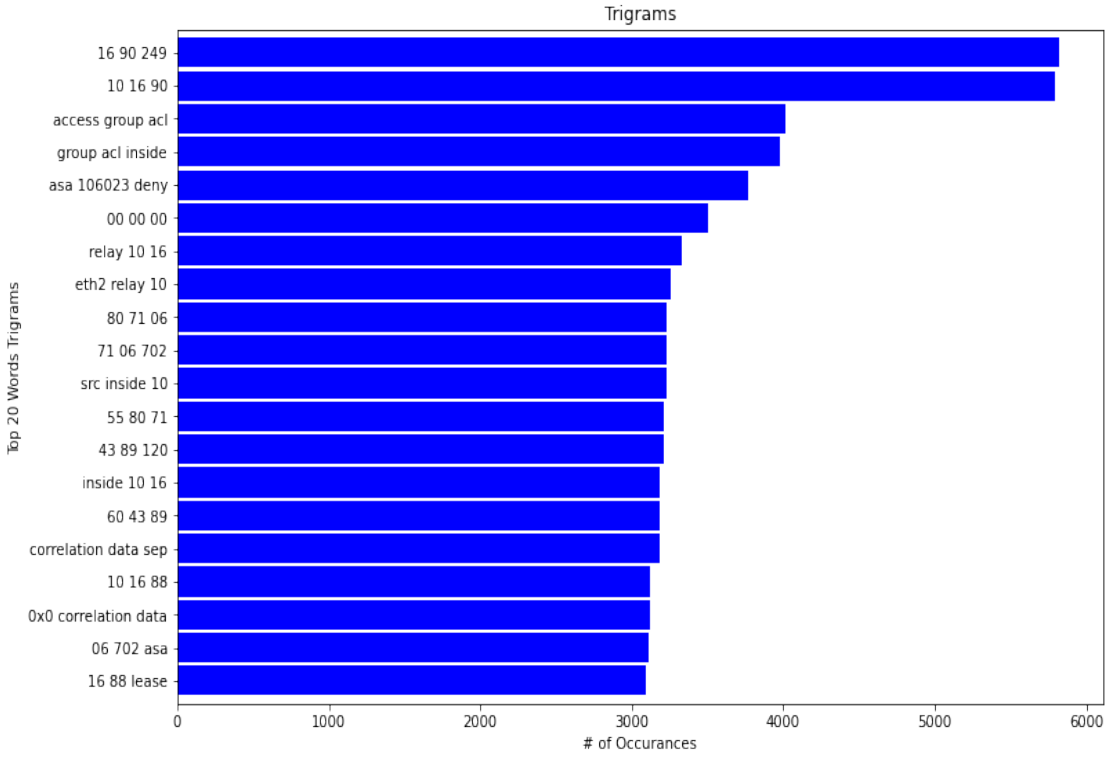
f. Top 20 Words:



g. Top 20 Bigrams



h. Top 20 Trigrams



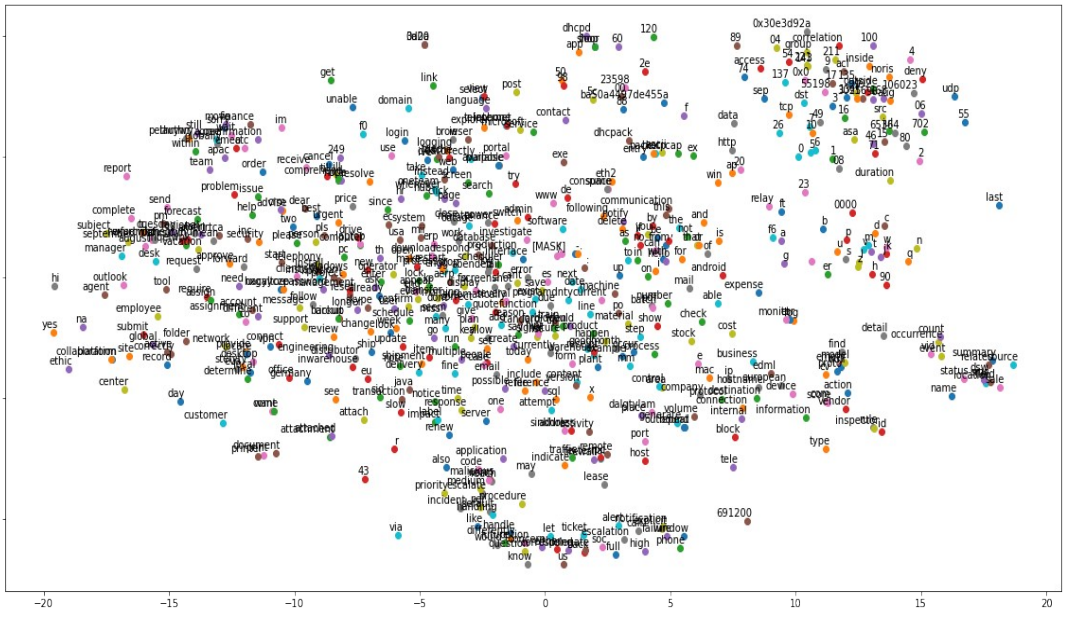
i. Word Embedding Visualization

To understand association between words in a large corpus of text, we have used Word2vec() and then plotted the same using t-SNE plots.

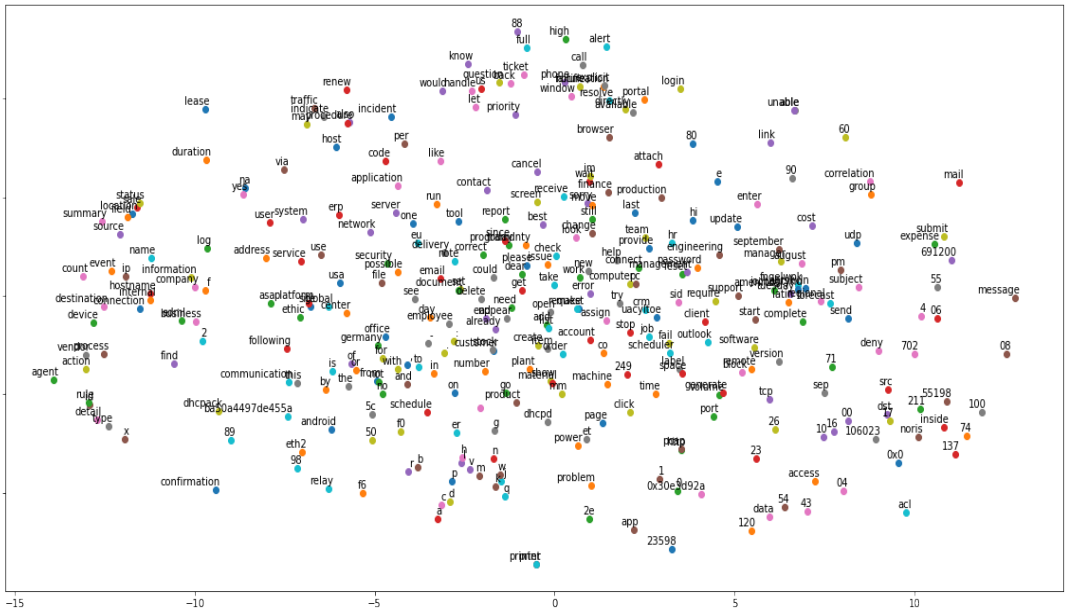
Word2vec is a technique for natural language processing. The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence.

t-SNE (tsne) is an algorithm for dimensionality reduction that is well-suited to visualizing high-dimensional data. The name stands for t-distributed Stochastic Neighbor Embedding. The idea is to embed high-dimensional points in low dimensions in a way that respects similarities between points.

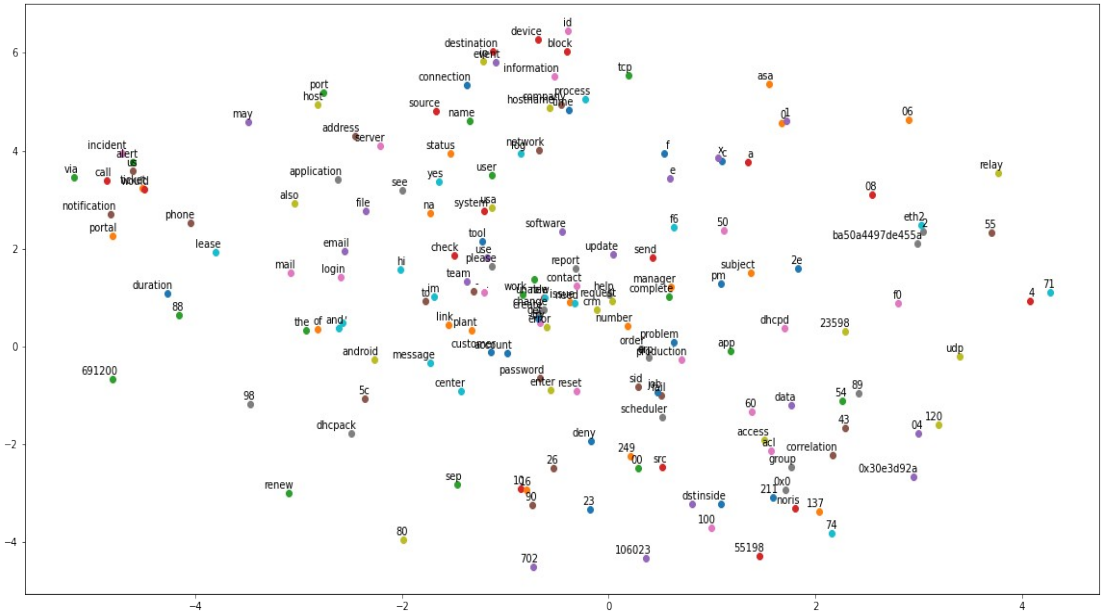
```
model = word2vec.Word2Vec(corpus, size=100, window=20, min_co
unt=500, workers=4)
```



```
model = word2vec.Word2Vec(corpus, size=100, window=20, min_co
unt=1000, workers=4)
```



```
model = word2vec.Word2Vec(corpus, size=100, window=20, min_co
unt=2000, workers=4)
```



4. Step by Step walk through of the solution

Data Cleansing

Following functions were applied on description and short description fields and created a new column Clean_Desc and Clean_Short_Desc.

a. **Clean_Sentence()**
   Removes special characters, greeting words like hello or email words like bcc/cc/to/subject/ sent etc. or email addresses from the sentence.
b. **Lemmatization and Stop Words Removal**
   Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.

   Stop words have been removed using nltk corpus module.

Language Translation

Both short description and description field seem to have German language words. Google Translator module was used to translate all German words to English language words. Both Clean_Short_Desc and Clean_Desc fields were passed through Google Translator service.

Combining Description

While working on cleansing of sentences and language translation it has been found that model accuracy can be increased if we combined Clean_Short_Desc and Clean_Desc fields. As a step, we created combined_sentence having all words from Clean_Desc fields and words from Clean_Short_Desc which are not found in Clean_Desc.

As these redundant columns are of not any use, we dropped short description, description, caller and Clean_Short_Desc fields from dataset and ultimately a new dataframe was created as clean_df.

Data Augmentation

As it was observed that data is highly skewed towards Grp_0 and rest groups having very minimal dataset, to increase classification model accuracy, we have used nlpaug modules for textual data augmentation.

We have used following augmentation functions/actions to generate/substitute more words for groups which have less than 60 tickets.

a. ContextualWordEmbdAug()

```
ContextualWordEmbsAug with bert-base-uncased and action as insert
e.g. Original:
The quick brown fox jumps over the lazy dog
Augmented Text:
even the quick brown fox usually jumps over the lazy dog
```

ContextualWordEmbsAug with bert-base-uncased and action as substitute

e.g. Original:

The quick brown fox jumps over the lazy dog

Augmented Text:

little quick brown fox jumps over the lazy dog

ContextualWordEmbsAug with distilbert-base-uncased and action as substitute
e.g. Original:
The quick brown fox jumps over the lazy dog .
Augmented Text:
the striped brown fox jumps over the muddy grass .

ContextualWordEmbsAug with roberta-base and action as substitute
e.g. Original:
The quick brown fox jumps over the lazy dog .
Augmented Text:
The quick brown fox jumps Into the bull dog .

b. Synonym Augmenter with augmentation source as "wordnet"

e.g. Original:
The quick brown fox jumps over the lazy dog .
Augmented Text:
The speedy brown fox jumps complete the lazy dog .

c. Random Word Augmenter with action as "Swap"

e.g. *Original:*
The quick brown fox jumps over the lazy dog .
Augmented Text:
Quick the brown fox jumps over the lazy dog .

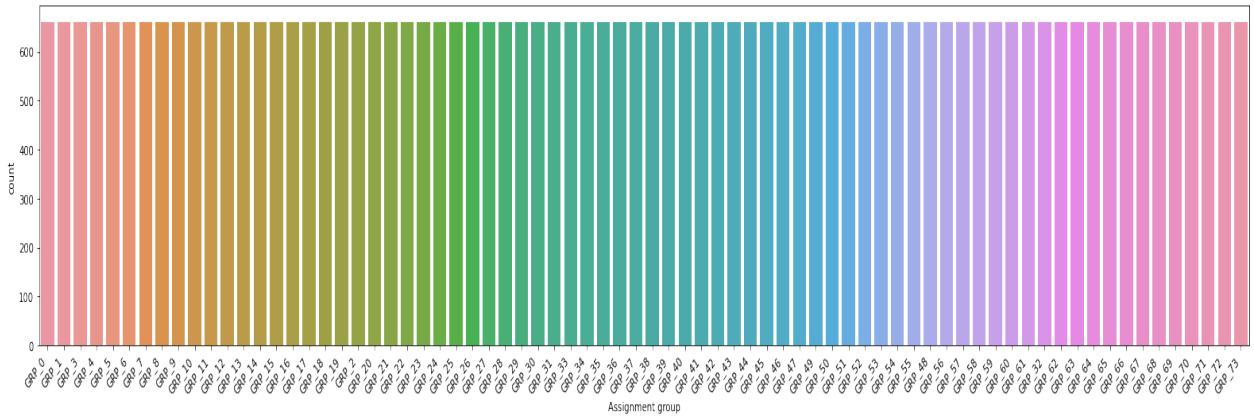
Even after augmentation, we find imbalance in the data and hence resampling was applied.

Resampling

As part of resampling exercise, we found that Grp_8 is having 661 tickets, highest amongst all the groups except Grp_0. It was decided to resample all the groups to have 660 samples.

Dataset now has total of 48914 rows with three columns now.

	combined_sentence	Assignment group	len_final_Description
7589	vpn connection issue connect user system use t...	GRP_0	18
7324	please access consultant vmhfteqo jpsfikow sch...	GRP_0	16
7358	sep encryption set	GRP_0	3
2791	need check payslip	GRP_0	3
2329	ticket update inplant	GRP_0	3



Now that data is balanced, we generated a few visualizations to get better understanding of the data.

Modeling

As the target classes are now balanced, we have applied following models to derive best model having highest accuracy.

We have used two Naïve Bayes classification algorithms of Machine Learning based on Bayes theorem which gives likelihood of occurrence of the event. Naive Bayes classifier is a probabilistic classifier which means that given an input, it predicts the probability of the input being classified for all the classes. It is also called conditional probability.

There are three types of Naive Bayes Classifiers out of which we have used first two.

a. **Multinomial Naive Bayes** - Widely used classifier for document classification which keeps the count of frequent words present in the documents.
b. **Bernoulli Naive Bayes** - Used for discrete data, where features are only in binary form.
c. Gaussian Naive Bayes - Used when we are dealing with continuous data and uses Gaussian distribution

Model 1: Tf-IDF vector with Multinomial Naive Bayes

Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. Whereas simple naive Bayes would model a document as the presence and absence of particular words, multinomial naive bayes explicitly models the word counts and adjusts the underlying calculations to deal with in.

TFIDF, short for term frequency–inverse document frequency, is a numerical statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

P(word|class)=(word_count_in_class + 1)/(total_words_in_class+total_unique_words_in_class)

Here, each word in each document is counted as per their TF-IDF weight. Then you get your probabilities for Naive Bayes using the formula highlighted above (adding up the TF-IDF weights instead of simply counting the number of words).

Model Accuracy:

Train Score	Test Score
89.16%	88.51%

Model 2: Glove Embedding with Bernoulli Naive Bayes

Word Embedding

As all our Machine Learning and Deep learning algorithms are incapable of processing strings or plain text in their raw form, word embeddings are 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.

We have experimented Glove embedding in our models with the dimension as 100.

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.

Model Building:

```
clf = BernoulliNB()
clf.fit(X_train, y_train)
pred_train = clf.predict(X_train)
print(accuracy_score(y_train, pred_train))
```

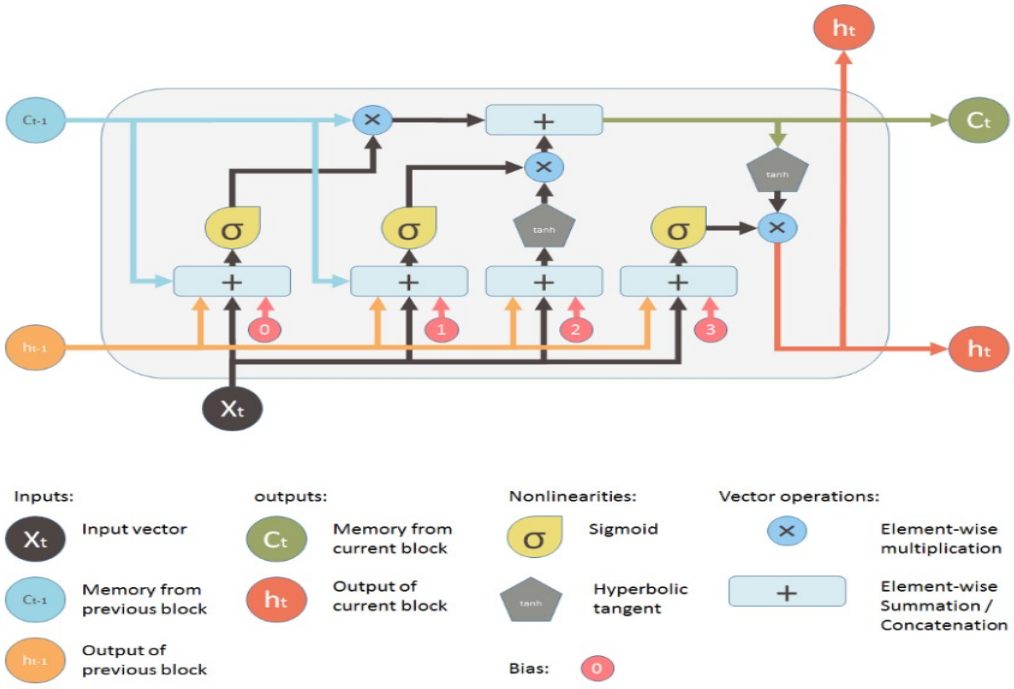
Model Accuracy:

Train Score	Test Score
61.67%	100%

Model 3: Glove embedding with simple LSTM model

Long short-term memory is an artificial recurrent neural network architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points, but also entire sequences of data.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs.



Here, LSTM network takes three inputs –

1) X_t is the input of the current time step.
2) h_t-1 is the output from the previous LSTM unit and
3) C_t-1 is the “memory” of the previous unit, which is the most important input.

Output contains:

1) h_t is the output of the current network.
2) C_t is the memory of the current unit.

Therefore, this single unit makes decision by considering the current input, previous output and previous memory. And it generates a new output and alters its memory.

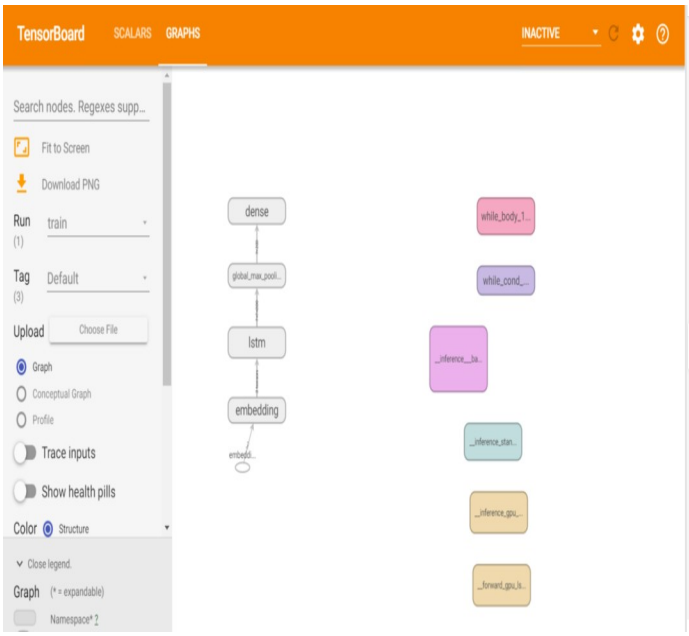
LSTM Model summary is as follows:

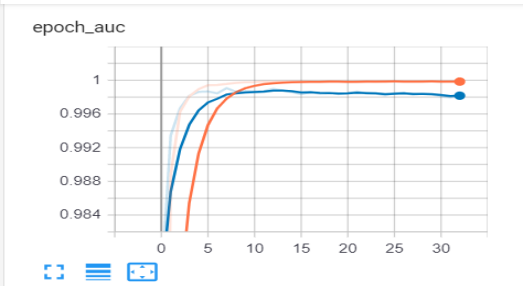
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 300)         3600300
_________________________________________________________________
lstm (LSTM)                  (None, None, 200)         400800
_________________________________________________________________
global_max_pooling1d (Global (None, 200)               0
_________________________________________________________________
dense (Dense)                (None, 74)                14874
=================================================================
Total params: 4,015,974
Trainable params: 4,015,974
Non-trainable params: 0
_________________________________________________________________
```

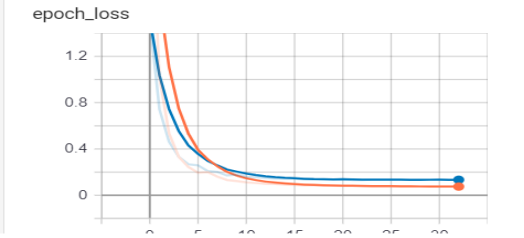
Model Accuracy:

Train Score	Test Score
97.46%	95.83%





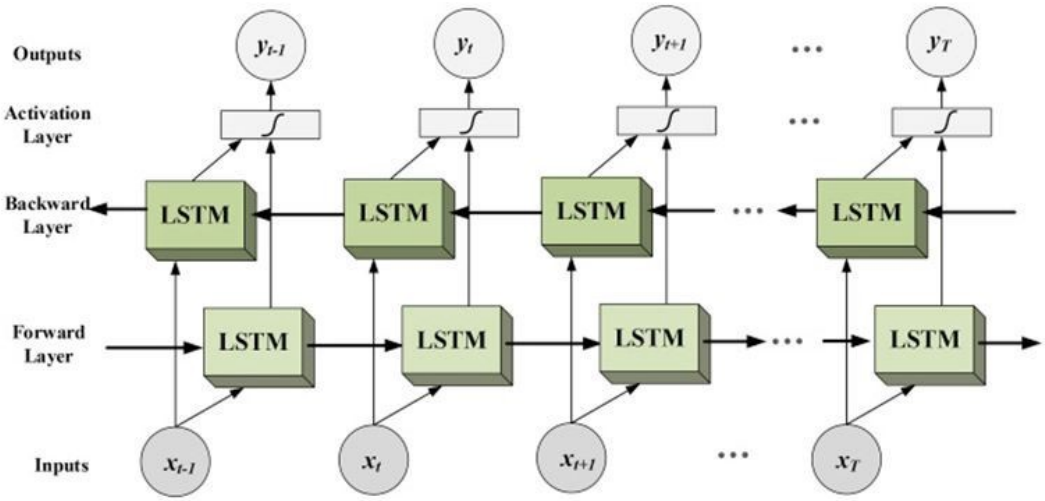
epoch_loss



Model 4: Glove Embedding with Bidirectional LSTM

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.

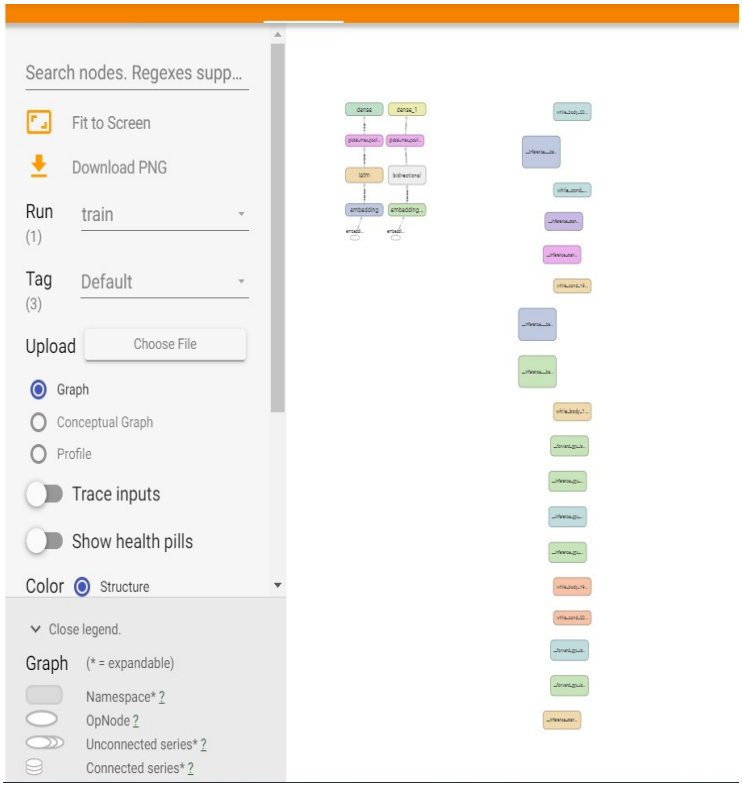


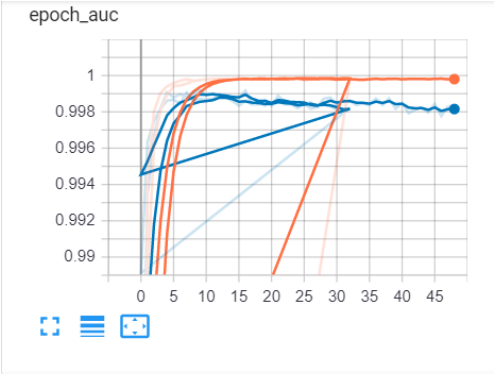
Bi-directional LSTM Model Summary:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 300)         3600300
_________________________________________________________________
bidirectional (Bidirectional (None, None, 480)         1038720
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 480)               0
_________________________________________________________________
dense_1 (Dense)              (None, 74)                35594
=================================================================
Total params: 4,674,614
Trainable params: 4,674,614
Non-trainable params: 0
_________________________________________________________________
```

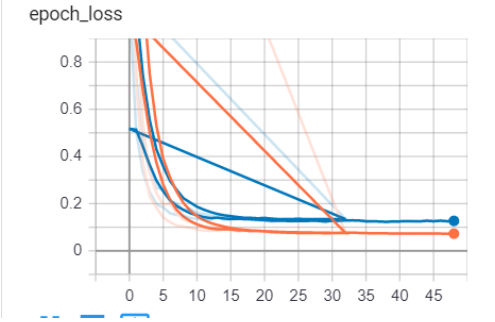
Model Accuracy:

Train Score	Test Score
97.19%	95.87%





epoch_loss



5. Model Evaluation

Summary of Training and Test Accuracy:

Model	Training Accuracy	Test Accuracy
Multinomial Naïve Bayes with Tf-IDF	89.16%	88.51%
Glove Embedding with Bernoulli Naive Bayes	61.67%	
Glove embedding with simple LSTM model	97.46%	95.83%
Glove Embedding with Bidirectional LSTM	97.19%	95.87%

We can see that LSTM and Bi-directional LSTM both gives almost the same accuracy. So, we would like to go with Bidirectional LSTM as it gives better accuracy when trained with raw data.

6. Comparison to Benchmark

From the given problem description, we could see that the existing system is able to assign 75% of the tickets correctly.

As our objective here is to build an AI-based classifier model to assign the tickets to right functional groups by analyzing the given description with an accuracy of at least 85%.

From the prediction results we see that the bi-directional LSTM model based on the resampled data is able to achieve an accuracy of 95.87% which is above our benchmark.

7. Implications

Although this model can classify the IT tickets with 95.87% accuracy, to achieve better accuracy in the real world, we would recommend having more than 500 datapoints for each of the groups.

8. Limitation

Except Grp_0, other groups are not having enough datapoints, we are not able to derive a model to reach 100% accuracy. Before applying this model into production, either additional datapoints are required from business team or additional manual intervention could help achieve higher accuracy.

9. Closing Reflections

Looking at the data points, looks like short description and description fields were manually entered and hence were having data entries with German language or with email addressed, special characters etc. and hence data cleansing becomes a critical task in entire predictability of target group.

To avoid this issue of data cleansing, it is recommended to have an unified port for raising such tickets e.g. My Support Portal which helps team to capture right amount of contextual data and ultimately helps NLP model to perform better in classifying ticket to correct group.