CAPSTONE PROJECT

Automation of Ticket Assignment (NLP)

May 2019 Batch (Group 10A)

Final Draft

*Refresh date is 17/***05/2020**

Team profile

Name	Description	Designation	email
Anuj Kumar Agrawal	Team Member	Director - Alight solutions	Ak0126002@gmail.com
Brijesh Kumar	Team Member	Director - Pumps	bksnsk11@hotmail.com
Girish Kumar	Team Member	Competency Center Lead - Shell	girish898@yahoo.com
Seema Malhotra	Team Member	Associate general manager - TCS	archieseema1@gmail.com
Raman Rangaswamy	Team Member	Enterprise Architect and Solutions Lead, tech Mahindra	rangaswamyraman@yahoo.com
Sidhanta Sekhar Maharana	Project Mentor	Faculty Member- Great learning institute	Sidhanta1989.maharana@gmail.com

Document Revision history

Version	Date	By	Description	Summary of Modifications
1.1	07/05/2020	Seema	Draft	Draft 1
1.2	11/05/2020	Seema	Draft	Detailed Summary, Models
1.1	07/05/2020	Brijesh	Draft	Visualization and summary, Track 2 details
1.2	11/05/2020	Anuj	Draft	Track 2 details
1.3	17/05/2020	Raman, Girish	Final	Format correction, Submission

Contents

1 Business Scenario

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.

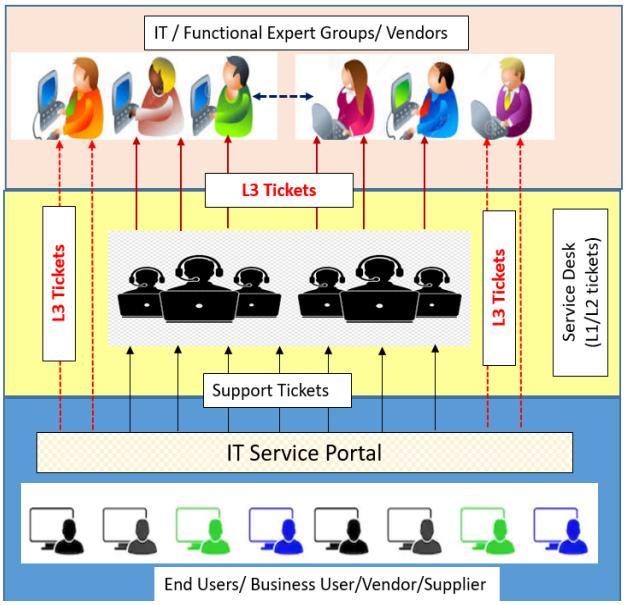
The main goal of Incident Management process is to provide a quick fix / workarounds or solutions that resolves the interruption and restores the service to its full capacity to ensure no business impact. In most of the organizations, incidents are created by various Business and IT Users, End Users/ 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.

2 Summary of problem statement

Let us try to reproduce the business scenario and key business values before we come to the problem statement.

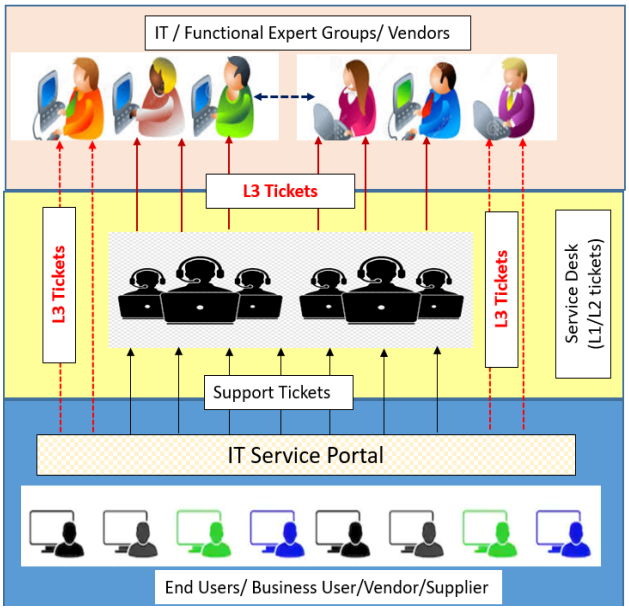
Current IT Support Process Setup:



IT Support Process :

- End users (composed of IT Users/ Business Users and Vendor and Supplier (if they have access to IT Support Portal) create IT incidents / tickets using IT Support Portal
- The tickets are automatically assigned to Service Desk.
- Service Desk resolves the L1/L2 tickets
- If Service Desk is unable to solve the tickets (L3 tickets), they assign the L3 tickets to Functional and IT Experts.
- Functional / IT Experts resolves the L3 tickets. If required, external vendor's help is taken for L3 tickets resolution
- L3 tickets/ priority tickets can be directly assigned to Functional experts in case of urgency via phone calls/ e-mails etc.

Keys Business Values:



Volumes being Handled :

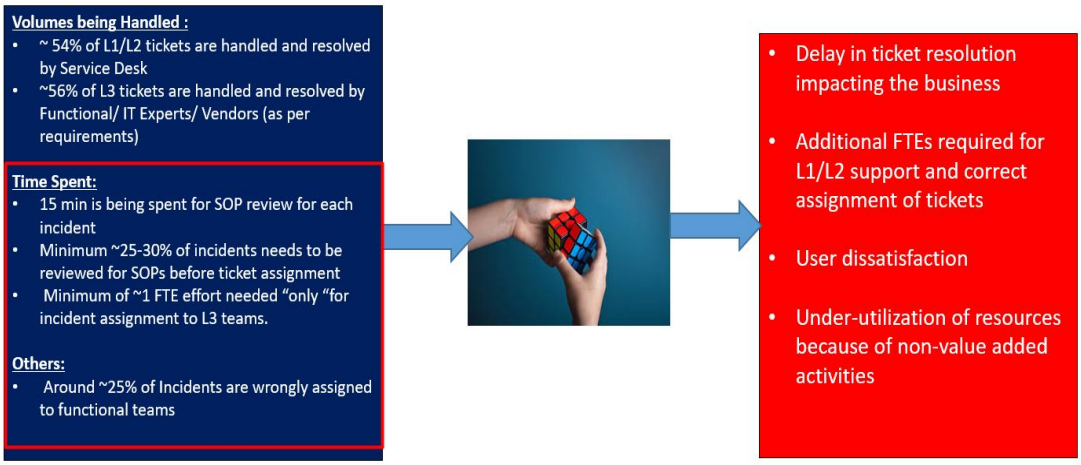
- ~ 54% of L1/L2 tickets are handled and resolved by Service Desk
- ~56% of L3 tickets are handled and resolved by Functional/ IT Experts/ Vendors (as per requirements)

Time Spent:

- 15 min is being spent for SOP review for each incident
- Minimum ~25-30% of incidents needs to be reviewed for SOPs before ticket assignment
- Minimum of ~1 FTE effort needed "only "for incident assignment to L3 teams.

Others:

- Around ~25% of Incidents are wrongly assigned to functional teams



With the above key business values, we can derive our problem statement as below**:**

If we look at the time spent below which are actually not for problem resolution but in administrative issues:

Time Spent for administrative issues:

- 15 min is being spent for SOP review for each incident
- Minimum ~25-30% of incidents needs to be reviewed for SOPs before ticket assignment
- Minimum of ~1 FTE effort needed "only "for incident assignment to L3 teams.

We can really see the "Problem areas":

- Around ~25% of Incidents are wrongly assigned to functional teams
- Delay in ticket resolution impacting the business
- Additional FTEs required for L1/L2 support and correct assignment of tickets
- User dissatisfaction and poor customer service
- Under-utilization of resources because of non-value-added activities

3 Data and findings (Exploratory Data Analysis)

First few records (observations) looks as follows:

	Short description	Description	Caller	Assignment group
0	login issue	-verified user details.(employee# & manager na...	spxjnwir pjlcoqds	GRP_0
1	outlook	\r\n\r\nreceived from: hmjdrvpb.komuaywn@gmail...	hmjdrvpb komuaywn	GRP_0
2	cant log in to vpn	\r\n\r\nreceived from: eylqgodm.ybqkwiam@gmail...	eylqgodm ybqkwiam	GRP_0
3	unable to access hr_tool page	unable to access hr_tool page	xbkucsvz gcpydteq	GRP_0
4	skype error	skype error	owlgqjme qhcozdfx	GRP_0
5	unable to log in to engineering tool and skype	unable to log in to engineering tool and skype	eflahbxn ltdgrvkz	GRP_0
6	event: critical:HostName_221.company.com the v...	event: critical:HostName_221.company.com the v...	jyoqwxhz clhxsoqy	GRP_1
7	ticket_no1550391- employment status - new non-...	ticket_no1550391- employment status - new non-...	eqzibjhw ymebpoih	GRP_0
8	unable to disable add ins on outlook	unable to disable add ins on outlook	mdbegvct dbvichlg	GRP_0
9	ticket update on inplant_874773	ticket update on inplant_874773	fumkcsji sarmtlhy	GRP_0

Last few records (observations) looks as follows:

	Short description	Description	Caller	Assignment group
8490	check status in purchasing	please contact ed pasgryowski (pasgryo) about ...	mpihysnw wrctgoan	GRP_29
8491	vpn for laptop	\n\nreceived from: jxgobwrm.qkugdipo@gmail.com...	jxgobwrm qkugdipo	GRP_34
8492	hr_tool etime option not visitble	hr_tool etime option not visitble	tmopbken ibzougsd	GRP_0
8493	erp fi - ob09, two accounts to be added	i am sorry, i have another two accounts that n...	ipwjorsc uboapexr	GRP_10
8494	tablet needs reimaged due to multiple issues w...	tablet needs reimaged due to multiple issues w...	cpmaidhj elbaqmtp	GRP_3
8495	emails not coming in from zz mail	\r\n\r\nreceived from: avglmrts.vhqmtiua@gmail...	avglmrts vhqmtiua	GRP_29
8496	telephony_software issue	telephony_software issue	rbozivdq gmlhrtvp	GRP_0
8497	vip2: windows password reset for tifpdchb pedx...	vip2: windows password reset for tifpdchb pedx...	oybwdsgx oxyhwrfz	GRP_0
8498	machine nÃ£o estÃ¡ funcionando	i am unable to access the machine utilities to...	ufawcgob aowhxjky	GRP_62
8499	an mehreren pc`s lassen sich verschiedene prgr...	an mehreren pc`s lassen sich verschiedene prgr...	kqvbrspl jyzoklfx	GRP_49

Basic information about the input data:

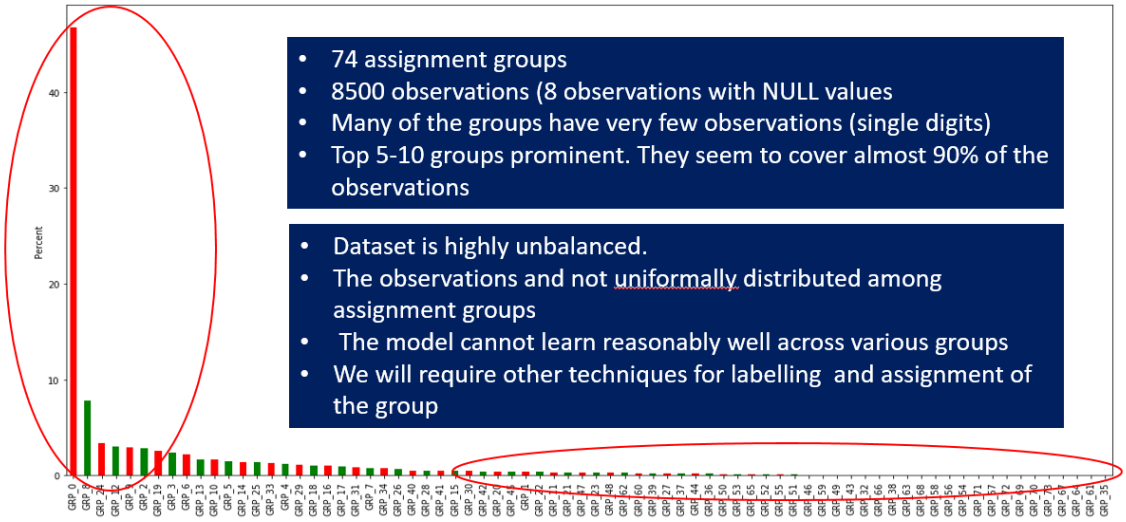
Data Descrption :

```
- Short description          object  *This is a shrt description of the issue/ problem (tickets)*
- Description                object  *This gives more detailed description of the tickets*
- Caller                     object  *This is the user caller who reported the issue*
- Assignment group           object  *This is the group to which the tickes has been assigned for resolution*
```

Shape of the data

```
- No of rows    = 8500
- No of columns = 4
```

Data Distribution



Summary of the data distribution

Observations:

- There are 74 assignment groups
- Group 0 seems to be the biggest accounting for about 45-50% of the total observations
- More than 50% of the groups seems to have only 1-2 observations
- 4 or 5 groups account for about 80% of the observations

Inference from data distribution:

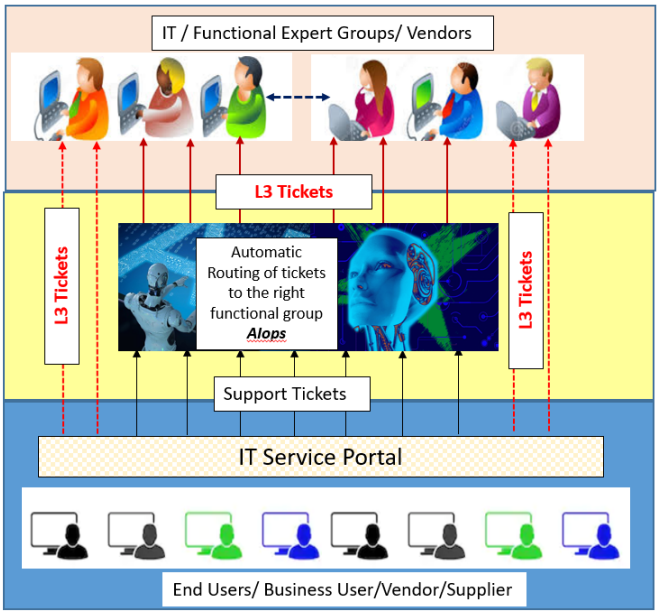
- Data is highly imbalanced based on Assignment Groups

- There will not be a good learning/ Training for all the Assignment groups because of very less observation for many of the groups

(We'll see it later that with manual trial and error of combining 74 groups to smaller groups with different algorithms and Keras model the accuracy was in between 45% to 65%)

4 Overview of the final process

This what we want to achieve for the organization:



Proposal :

- To automate the ticket the process of ticket analysis and assignment to the right functional / IT expert groups (using Machine Learning / AI tool)

Objective:

- To reduce the ticket assignment time
- To reduce ticket resolution time
- To reduce wrong assignment of tickets
- To optimize the resources

e.g. Service Desk can utilize the time for ticket resolution instead of going through SOP and ticket assignment

ML/AI can work 24X7 through out the year (without worrying about corona ☺)

We can summarize the objectives as follows:

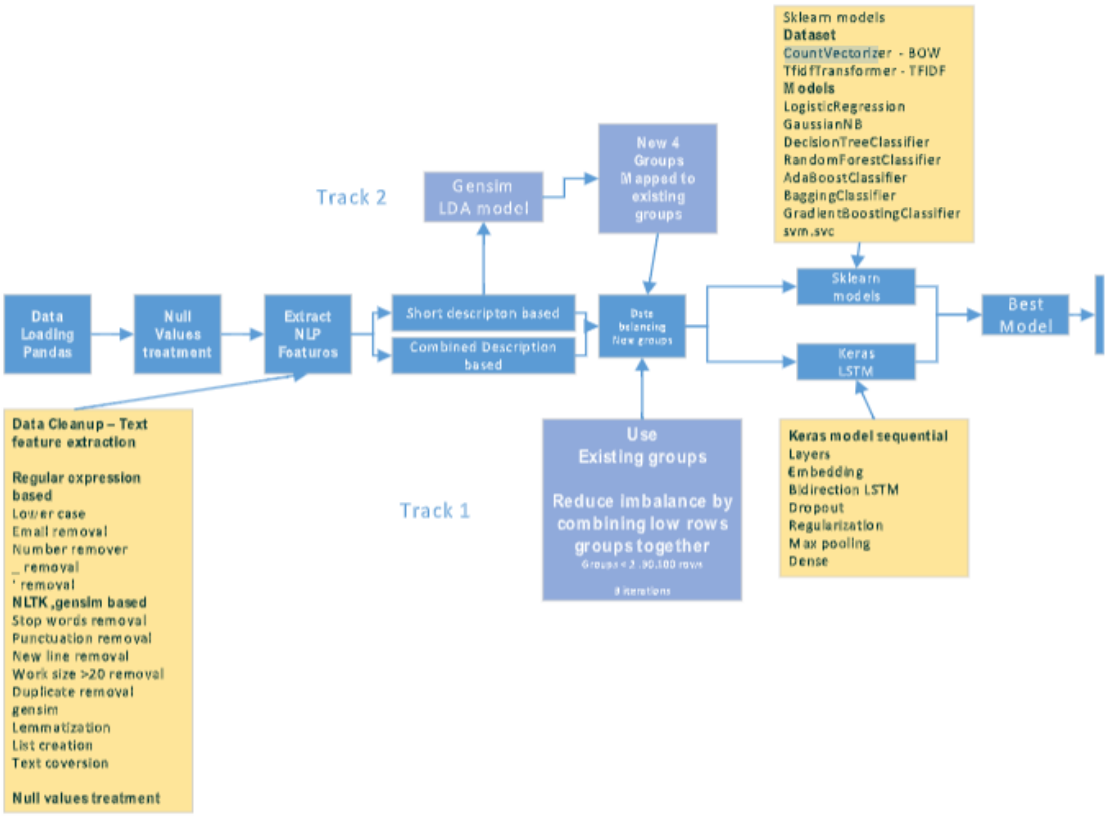
- To reduce the ticket assignment time
- To reduce ticket resolution time
- To reduce wrong assignment of tickets
- To optimize the resources

e.g. Service Desk can utilize the time for ticket resolution instead of going through SOP and ticket assignment

ML/AI can work 24X7 throughout the year (without worrying about corona)

4.1 Description of our solution methodology:

The under depicted diagram depicts our approach/ methodology to solve the problem**:**



For Pre-processing of the data we took a usual Approach of Data loading in Pandas , Missing value treatment and then extracting NLP features .

Our methodology follows 2 Tracks:

- Track 1 – Using Existing groups (Multi iteration approach by reducing imbalance among group)
- Track 2 – Genism LDA Model

Further, both tracks use Multi models approach to find the best accuracy–

- Model 1 – Sklearn Models (around 6 core modules)
- Model 2 – Keras Sequential (Using TensorFlow)

Finally, we processed the analysis to depict the best model in both approaches

Data/ Text Pre-processing Steps:

Before we start working on any of the Machine Learning / Deep Learning / Natural Language processing, there are some general step we should follow to clean the text data. General Steps are:

Data/Text Cleaning

- Missing Value Treatment (Dropping null Values)
- Convert the text to lower case
- Remove digits, punctuation marks, new line, carriage return and unwanted spaces
- Remove stop words (stop words are the words which are filtered out before or after processing of natural language data. Though "stop words" usually refers to the most common words in a language, there is no single universal list of stop words)
- Tokenization, Stemming and Lemmatization

Data/Text Pre-processing

- Creation of Dictionary of words
- Feature Extraction (Vectorization)
- Corpus Preparation

Algorithm Used

For features creation:

- Bag-of-words (BoW)
- TF-IDF (Term Frequency-Inverse Document Frequency)
- GloVe (Global Vectors)

For Topic Modelling:

- Latent Dirichlet Allocation (LDA)

SkLearn Algorithms (Classifiers)

- Linear Regression
- Decision Tree
- Random Forest
- AdaBoost
- GradientBoost
- Multinomial Naïve Bayes
- Support Vector

Deep Learning

- Keras : Bidirectional LSTM

Combining Techniques:

Need of feature extraction techniques:

The main problem in working with language processing is that machine learning algorithms cannot work on the raw text directly. So, we need some feature extraction techniques to convert text into a matrix (or vector) of features. We used Bag-of-words and TF-IDF for LDA and for Sklearn algorithms. We used GloVe for Keras LSTM Model

Need of LDA (Latent Dirichlet Allocation) :

As we have seen previously that with 74 groups, we could not really justify a model which can be used to AUTOMATICALLY assign the tickets to a group with accuracy better than the current ITSP process. Manual trial and error of regrouping didn't help either.

We used Latent Dirichlet Allocation (LDA) modelling technique classify text in a document (data sets). LDA builds a topic per document model and words per topic model, modeled as Dirichlet distributions.

Sklearn and Keras LSTM:

We used various classifiers to see if our final model using LSTM is be generalized. The classifiers scores also set up a benchmark we should try to achieve by using LSTM deep learning network. If different classifiers scores are comparable, we can say that our model will perform in the same way (similar accuracy) in production environment.

5 Step-by-step solution walk through

5.1 Model code files

Please node - This model works on different iteration to arrive at optimal number of Level Groups so as to find the best accuracy – based on optimal group setting and ease of understanding the approach – **Step 4 is divided into 2 tracks**. Rest of the coding is almost same

For Track1 –

Capstone_Group10A_track1_milestone3.ipynb
Capstone_Group10A_track1_milestone3.html

For Trace 2 –

Capstone_Group10A_track2_milestone3.ipynb
Capstone_Group10A_track2_milestone3.html

Step 1: Data Exploration

Main purpose of this step is to see and explore the existing data sets. Important activities in this step are as follows:

- Data Distribution
- Data Types
- Features/ Attributes
- NULL Values

Step 2: Data / Text Cleaning:

This is one of the most important steps. Our model's performance is mainly dependent on the quality of the data on which it is trained. Almost 70-75% of the time of the whole project goes into data cleaning and data preparation for our model building

Important activities in this step are as follows:

- NULL Values Treatment
- Covert to Lower Case
- Remove Punctuation Remove Digits
- Remove E-mail IDs, URL
- Remove Stop Words

By doing the above steps, we remove those characters / words which are not required for our model building. They are noise in our datasets and should be removed. This step provides us the clean text which will be used by next steps to tokenize sentences/ words and vocabulary (dictionary of words), feature extraction and corpus preparation

Step 3: Data / Text Pre-processing:

As we have mentioned earlier, even though we have the clean data / text from step 2, it cannot be used by any algorithm. The text is of different lengths and cannot be fed to any of the algorithm or model. To make the text compatible for our model, we will have to convert the text into matrix / vectors. This process is called feature extraction. We completed the following activities in this step

- Tokenization
- Dictionary Creation
- Feature Extraction: BoW, TF-IDF, GloVe
- Corpus Preparation

Step 4: Remove imbalance from Group

Track 1 - Use existing group – Multiple iteration

Reduce number of groups

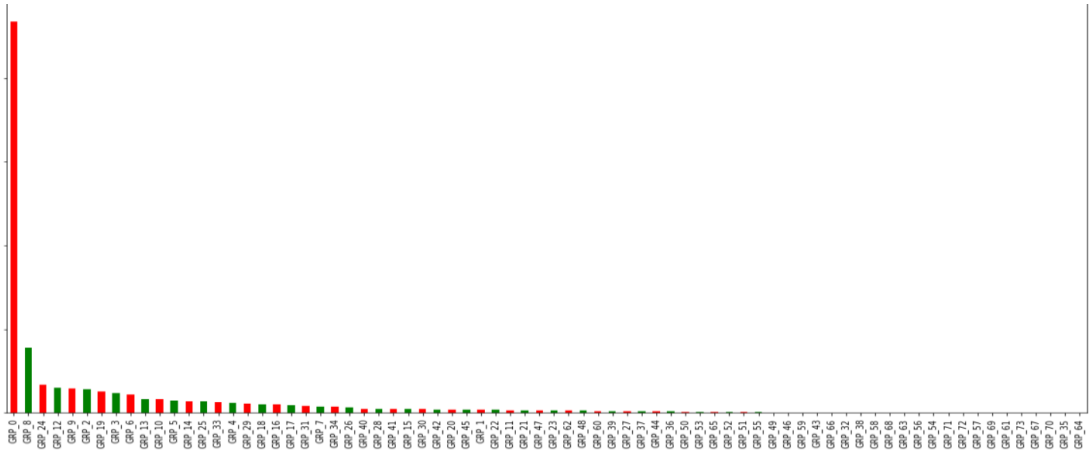
- Combine groups with only 1 row to a new Group 74
- Combine Groups with <30 rows to a new Group 74
- Combine groups with <100 rows to a new group 74

Track2 – LDA modeling to find out entirely new groups and reduce imbalance

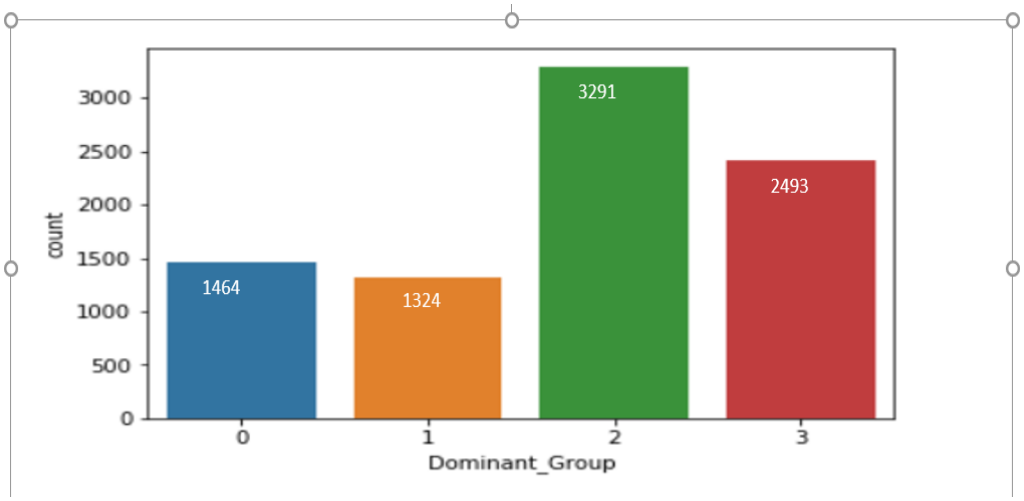
Reduce the number of groups using LDA modeling and find out best number of new groups

This was not a mandatory step but it became very important step for us. As we have seen, we had 74 Assignment Group in our initial data sets. After completing all steps (step 1 to step 3) when we tried to make the model with original 74 groups as well as manually creating lesser number of groups(e.g. 40, 30, 20 etc.) , our accuracy ranged between 45% - 65% , which was MUCH LOWER than the existing accuracy level (manual assignment which is about 75%). Hence there was a need to have an automatic and statistical way to Regroup the existing Assignment Groups.

We used LDA algorithm (which is used for topic modelling). LDA takes the vectorised corpus and tried to create the related groups based on the topic. We used it as an Unsupervised learning algorithm. We passed on the inputs data and expected LDA to provide us the groups. After few trials we could divide the dataset in 4 NEW Assignment groups



Pic:Data Distribution (74 assignment groups, highly unbalanced)



Pic: Data distribution (4 Assignment groups, much better than before) after LDA

Step 5: Sklearn algorithms for model evaluation:

As we have new / updated data set, which was cleaned, pre-preprocessed regrouped in 4 groups, we were ready for out model creation.

Before trying out our Deep Learning Algorithm (LSTM), we used various classifiers to get a very good idea about the model performance. We used the following classifiers

- Linear Regression
- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boost

- Multinomial Naïve Bayes
- Support Vector

We also used both Bag-of-Words and TF-IDF vectors for all the above algorithms. For Decision Tree and SVM-SVC, we also used hyperparameter tuning using GridSearchCV and **RandomizedSearchCV** to get the best parameter by hyperparameter tuning .

The accuracy scores ranged between 80%-90% approximately. (see visualization)

This also set up the BENCHMARK SCORE for our final Model using Keras LSTM

Step 6: Final model (Keras LSTM):

Here we used Keras LSTM model (different layers, see visualization). In this case, we didn't use Bag-of-Words or TF-IDF vectors. Both these matrix / vectors create Sparse Matrix. We used the concept of Word Embeddings which reduces the Sparse Matrix. Word Embedding can be compared with PCA (Principal Component Analysis) which reduces the no. of features without compromising much on the accuracy.

We used GloVe (Global Vectors) for word embedding and used pre-trained word embeddings.

6 Model evaluation

We used different scores to evaluate the performance of our final model (Keras LSTM). We used accuracy score, precision score, recall score, F1-score and ROC AUC Score. The objectives was to mainly ensure that we have a model which is accurate also as well as generalized also (means it should work equally well in test and production environment). We also compared the scores with other classifier score to validate our model accuracy.

We tried out 100, 200 and 300 words embedding (pre-trained word embeddings)

Parameters which were important for us were:

- Vocabulary Length / maximum features
- Maximum lengths of the sentence
- Embedding Sizes (100, 200,300)

We also made use of No. of epochs, batch size and combination of deep learning layers to generalize the mode as well as to increase the accuracy.

We also used the concept of call backs (Model Check point, Early Stopping, Learning rate). We also tried out different losses (cross entropy, MSE loss etc.).

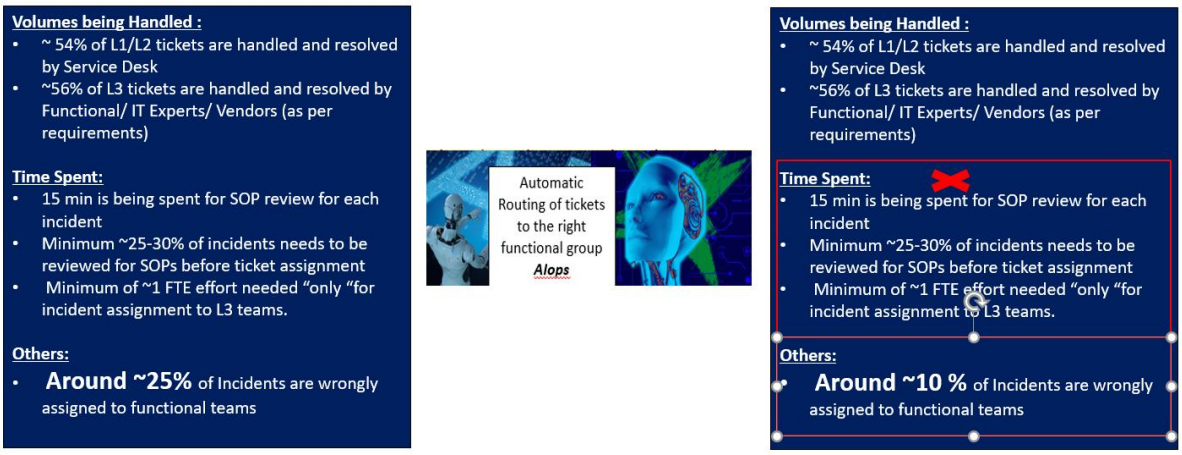
We also tried adam and sgd optimizer and finally settled for "adam" optimizer

For the final version, we used 25 epochs (We can see the visualization to see the effect of accuracy/loss vs. no. of epochs) to fine tune the final model

Comparison to the Benchmark:

Our benchmark, of course was to reach at about 85% (which is about 10% better than the existing accuracy of 75% in ticket assignment). We could reach upto that. We could reach to about 87-88% accuracy (We can take 85% as average with 95% confidence level)

(Initially, we did not even imagine it . We felt that we may come close to 75% but may not reach it)



7 Visualizations

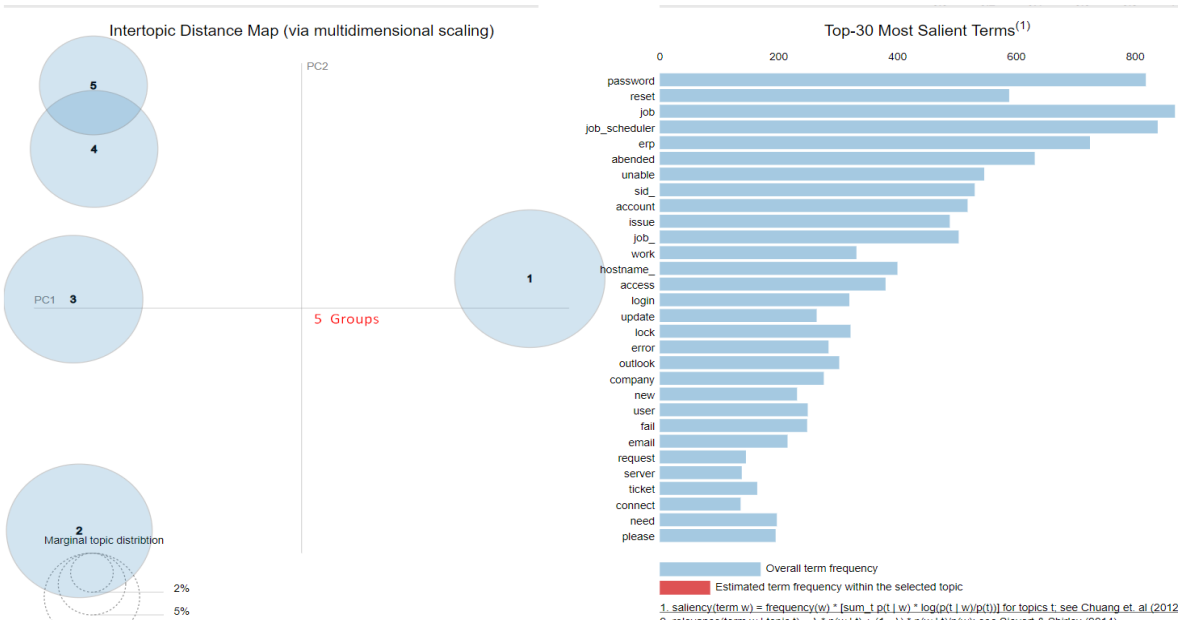
7.1 LDA Visualization by pyLDAvis package

You might be wondering, as to why we reached to a magic number of 4 groups from topic modelling using LDA.

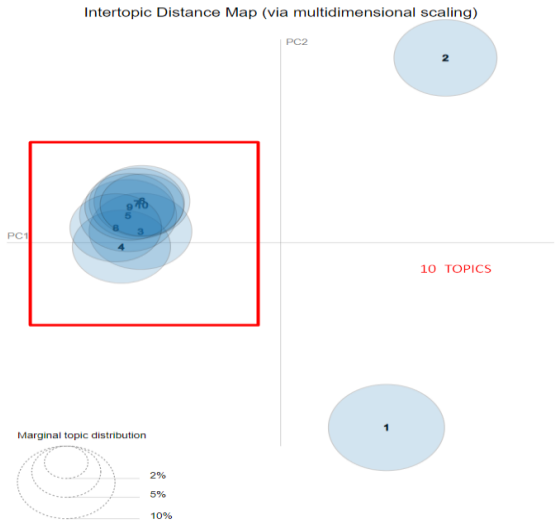
Actually, there was no magic. It was to play around with the parameter 'num_topics' to see if we can get distinctly apart circles (bubbles)

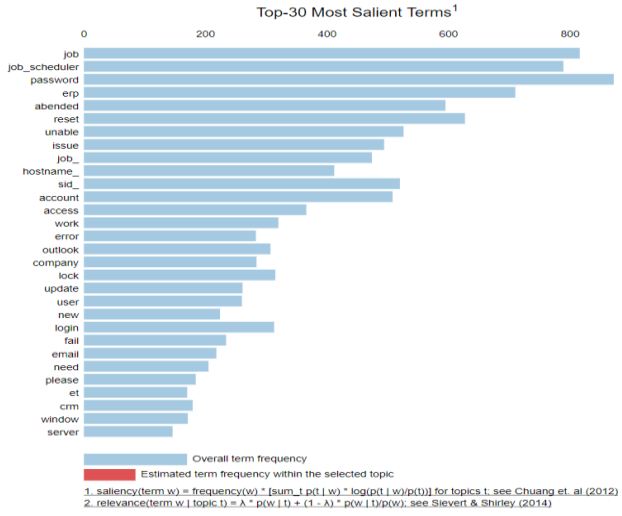
```
## Let us create functioon for LDA . We will use it for both Bag-of-word and TFIDF
def generate_lda_model(input_corpus):
    lda_model = gensim.models.ldamodel.LdaModel(corpus= input_corpus,
                                                id2word=dictionary,
                                                num_topics=4,      ## We tried different values for this hyperparameter
                                                random_state=100,
                                                update_every=1,
                                                chunksize=100,
                                                passes=10,
                                                alpha='auto',
                                                per_word_topics=True)
    return(lda_model)
```

Here the num_topics = 5 and we can see some overlap between bubble 4&5

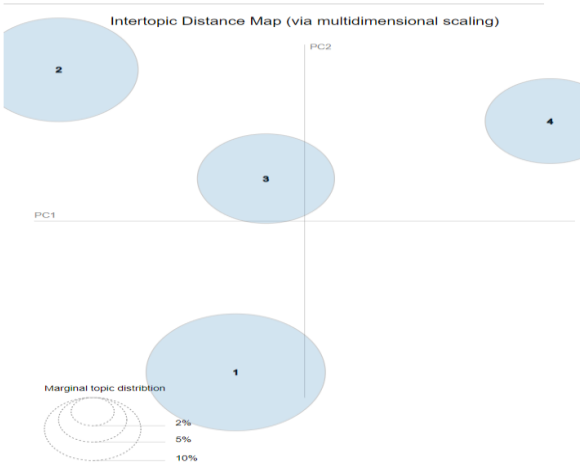


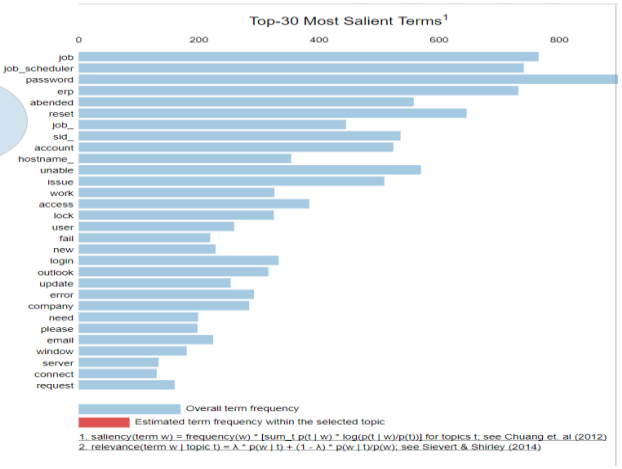
Here the num_topics = 10 and we can see some overlap between bubble except 1& 2

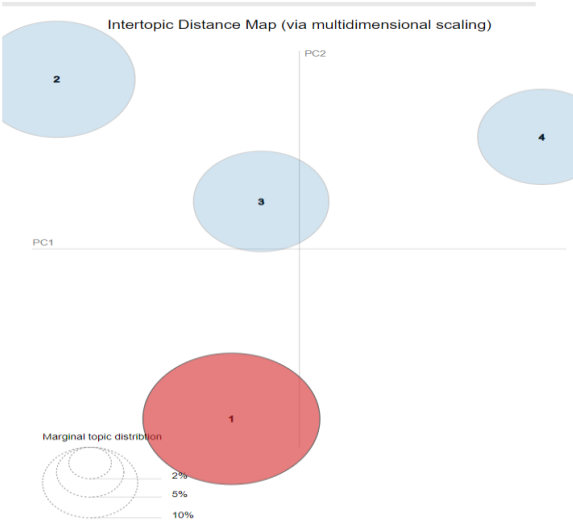


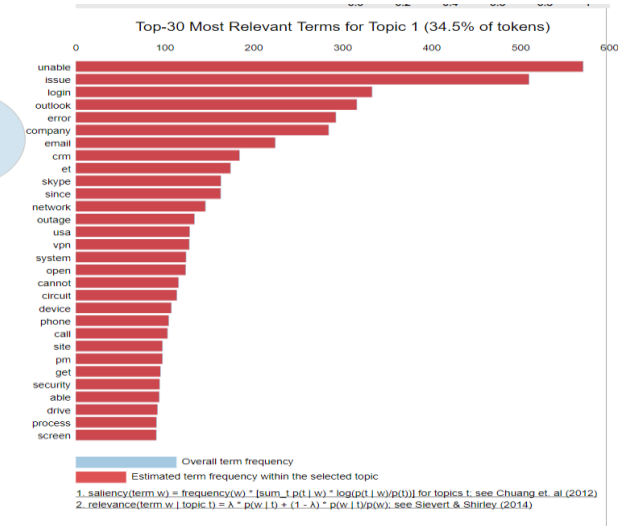


Here the num_topics = 4 and we can see no overlap (corpus= Bag-of_words).This is an ideal situation

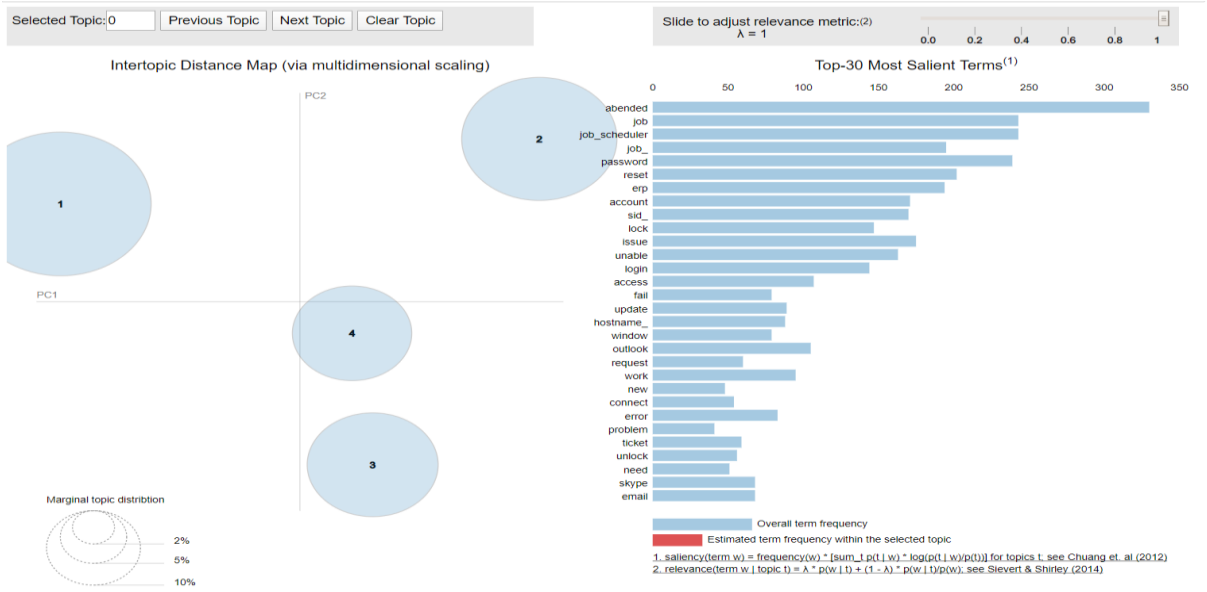


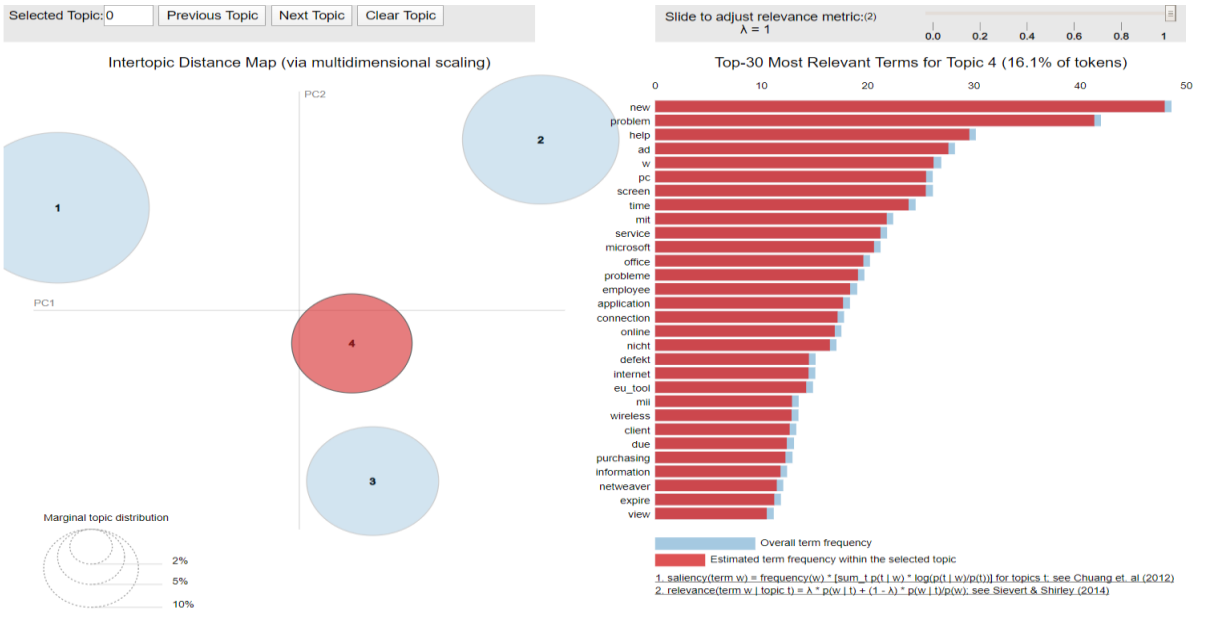






Here the num_topics = 4 and we can see no overlap (corpus= TF-IDF).This is an ideal situation





Interpretation of LDA Model

For both Bag-of-words and TF-IDF corpus, we can see some circle/bubbles (4 in our case) and on the right side a bar graph.

- Number if Circled define the group/ topics (which we have given as an input parameter and which is a hyperparameter also)
- The lesser is the intersection between the circles the better it is for us. No intersection is a boon If the distinctive circles are far apart (distance between the circles), it is an icing on the cake.
- What it indicates is that we based on the corpus, we can have 4 DISTINCT Topics
  If the circles are overlapping , we don't have really distinct topics based on the corpus.

That is the reason we play around with diffrent values of num_topics parameter (3-15) and we got the best result with num_topics = 4

- The bar chart on the right provides the 30 top words in a topic. If we click on any circle we'll see the result changing on right side (red color) Blue colour provides overall term frequency under the topic and red colur provides the estimated frequeny within the selected topic.

Here also we'll see a very good aount of overlap / fit

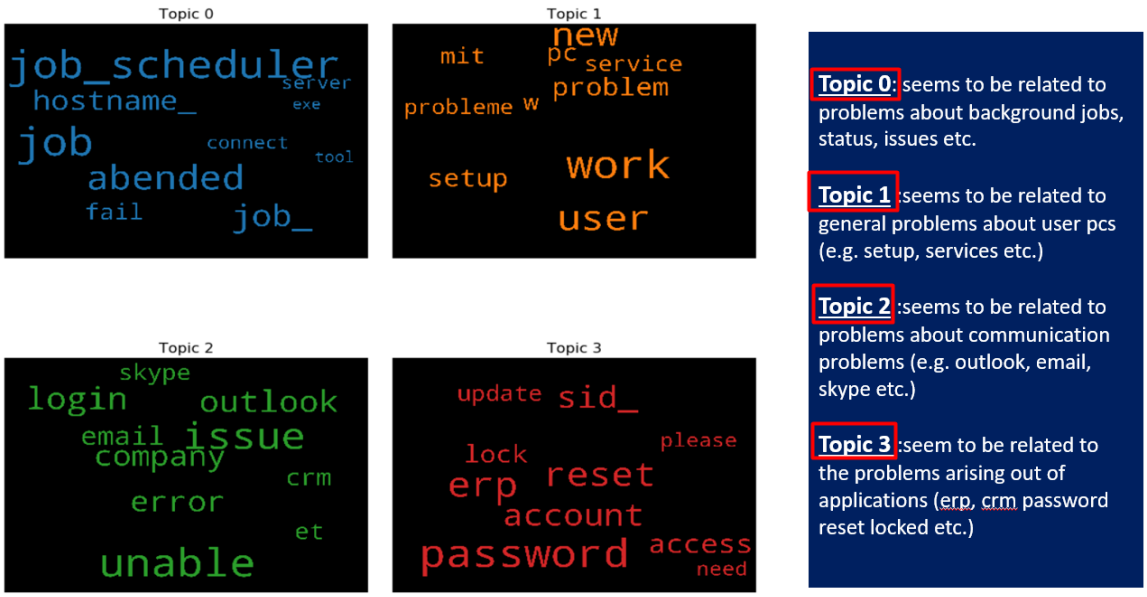
Also both the LDA model (with Bag-of-words and TF-IDF) gives similar results which gives us the confidence that the topics created can really be generalized for our modelling

7.2 Word Cloud Visualization

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud

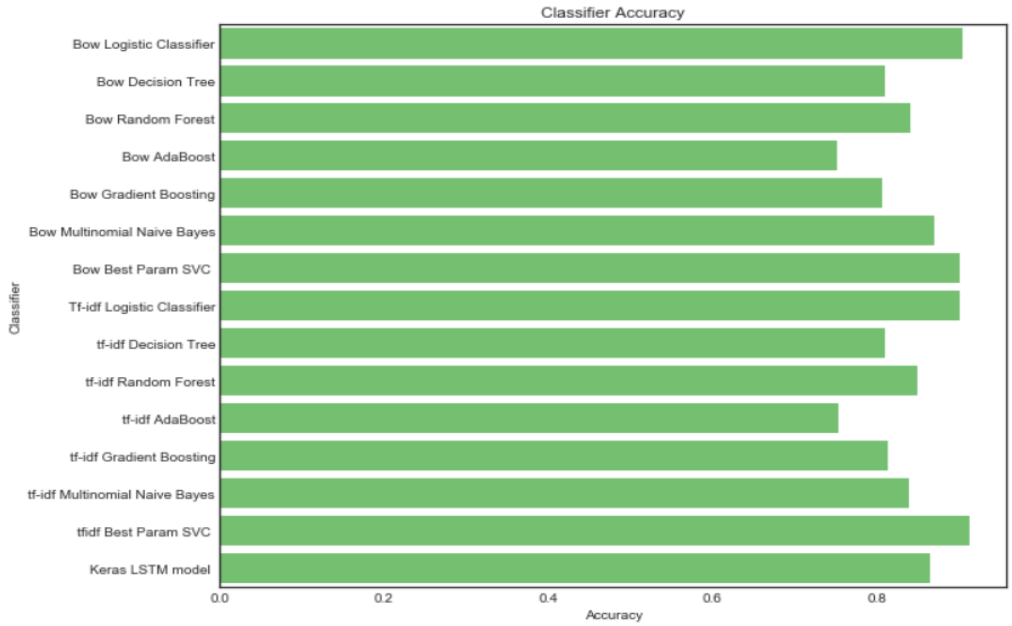
Word Cloud (using Bag-of-words corpus)

Word Cloud (using TF-IDF corpus)

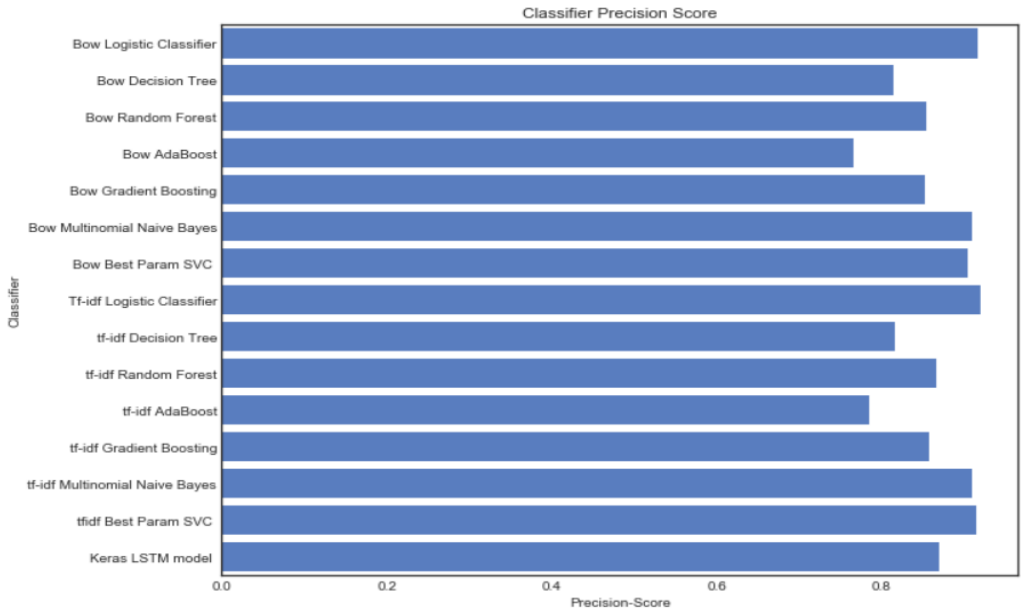


7.3 Score Comparison

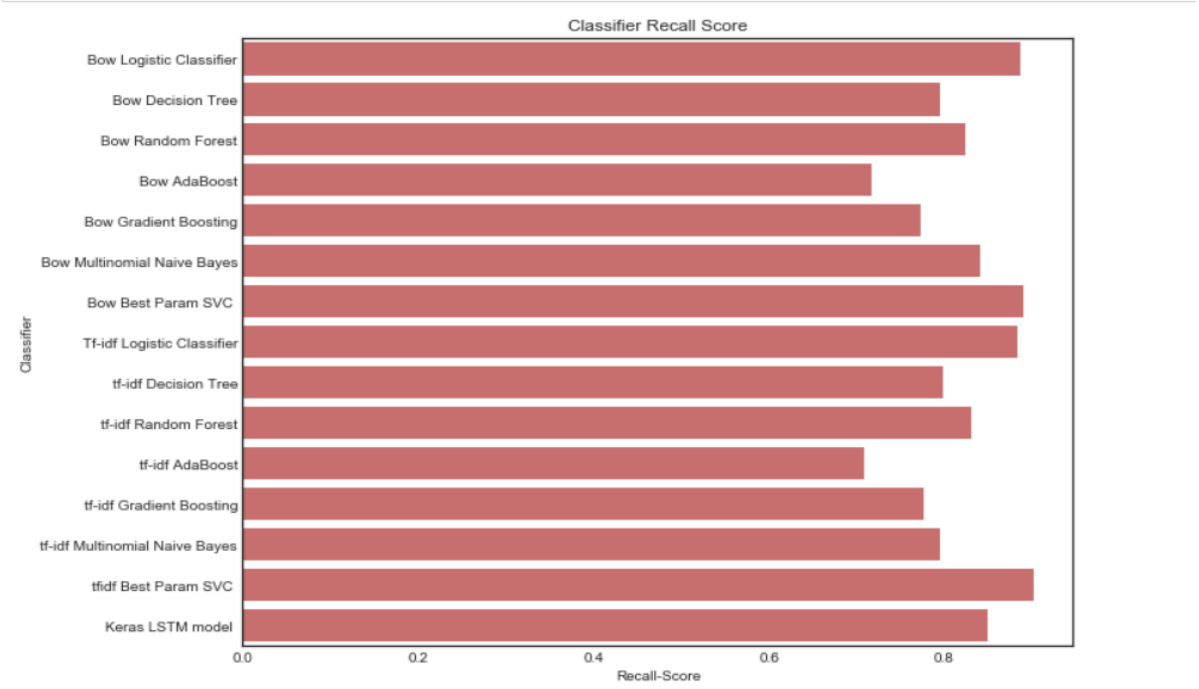
Classifier Accuracy Score:



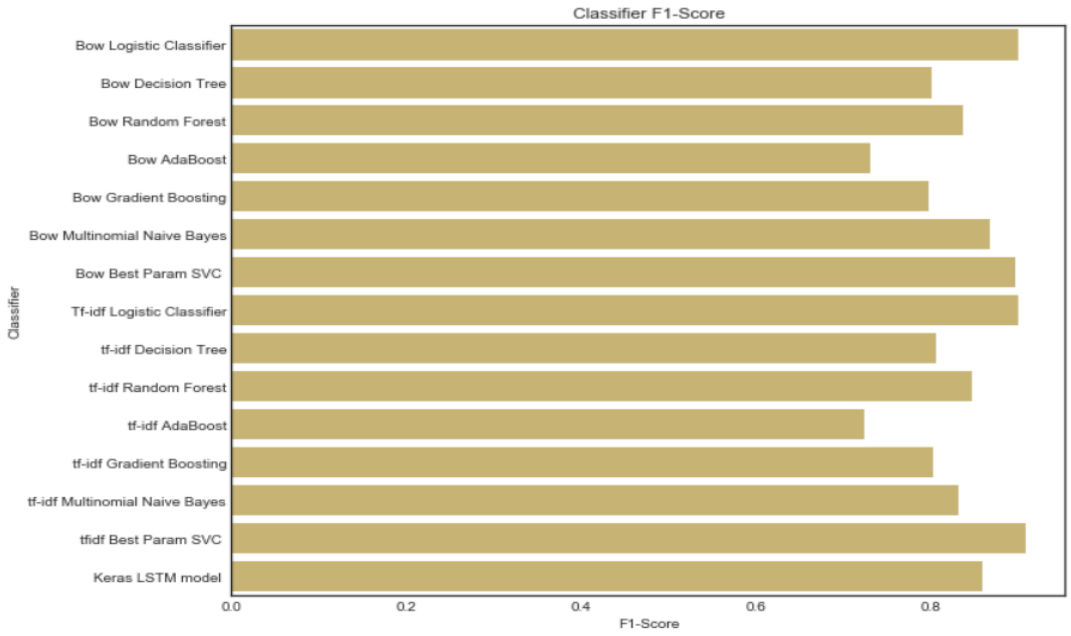
Classifier Precision Score:



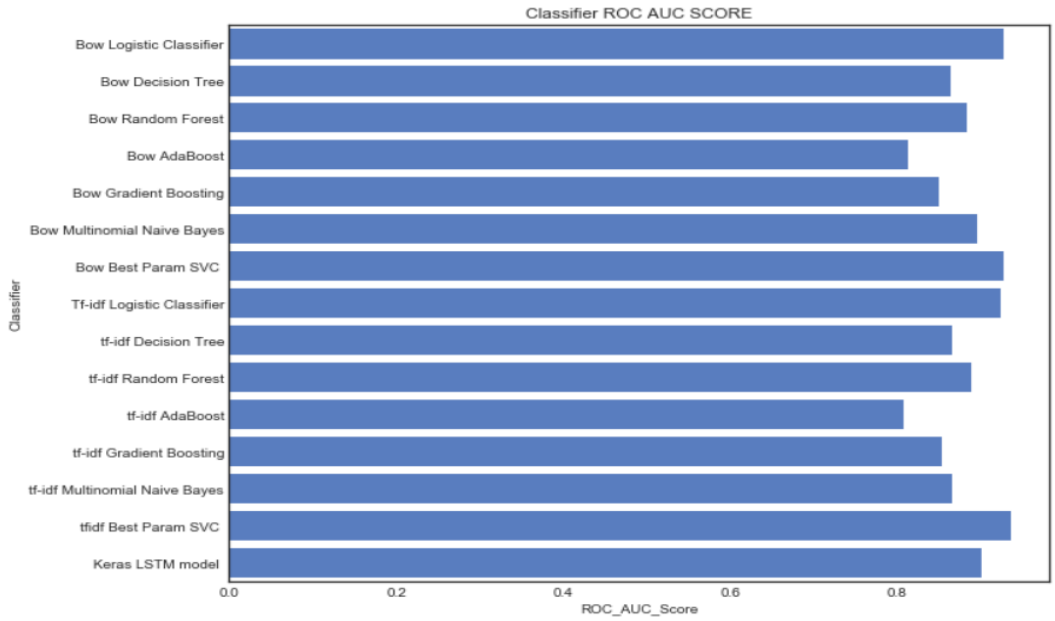
Classifier Recall Score:



Classifier F1-Score Score:



Classifier F1-Score Score:



$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN},$$

$$\text{Precision} = \frac{TP}{TP + FP},$$

$$\text{Recall} = \frac{TP}{TP + FN},$$

$$\text{F1 score} = 2 * \frac{Precision * Recall}{Precision + Recall},$$

True Positive (TP) denotes the number of real positives among the predicted positives,
True Negative (TN) denotes the number of real negatives of the predicted negatives.
False Negative (FN) denotes the number of real positives among predicted negatives
False Positive (FP) denotes the number of real negatives among predicted positives.

Accuracy denotes the proportion of documents classified correctly among all documents
Recall denotes the proportion of documents that are classified as positive among all real positive documents. Precision denotes the percentage of documents that are real positive among documents classified as positive
F1 score denotes the average of the weighted recall and precision scores

Final Keras Model (using LSTM):

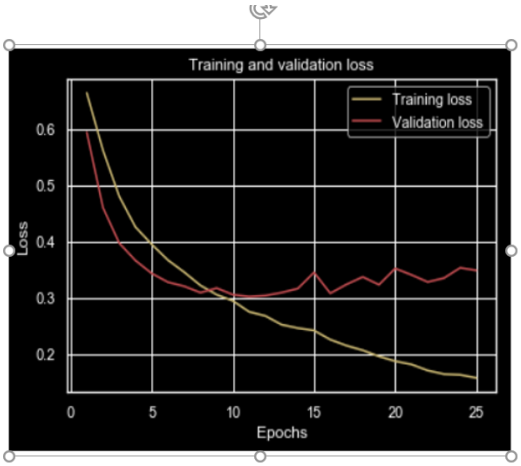
Keras LSTM Deep Learning Model

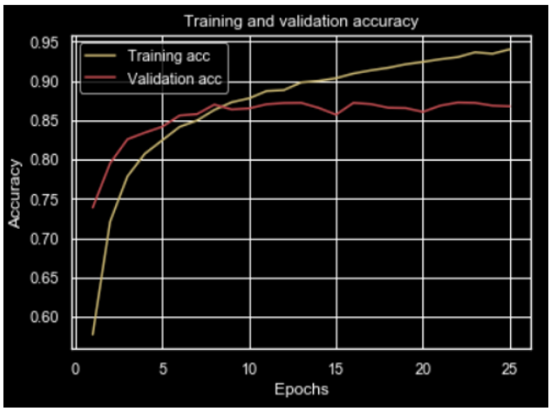
Here is the model summary

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, None, 200)	1059000
spatial_dropout1d_2 (Spatial	(None, None, 200)	0
bidirectional_2 (Bidirection	(None, None, 200)	240800
global_max_pooling1d_2 (Glob	(None, 200)	0
dense_5 (Dense)	(None, 128)	25728
dropout_4 (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 64)	8256
dropout_5 (Dropout)	(None, 64)	0
dense_7 (Dense)	(None, 32)	2080
dropout_6 (Dropout)	(None, 32)	0
dense_8 (Dense)	(None, 4)	132

Total params: 1,335,996
Trainable params: 276,996
Non-trainable params: 1,059,000

Performance of Loss and Accuracy v/s no. of epochs):


Training and validation loss
Training loss
Validation loss
Loss
0.6
0.5
0.4
0.3
0.2
0
5
10
15
20
25
Epochs


Training and validation accuracy
Training acc
Validation acc
Accuracy
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
5
10
15
20
25
Epochs

8 Model Deployment

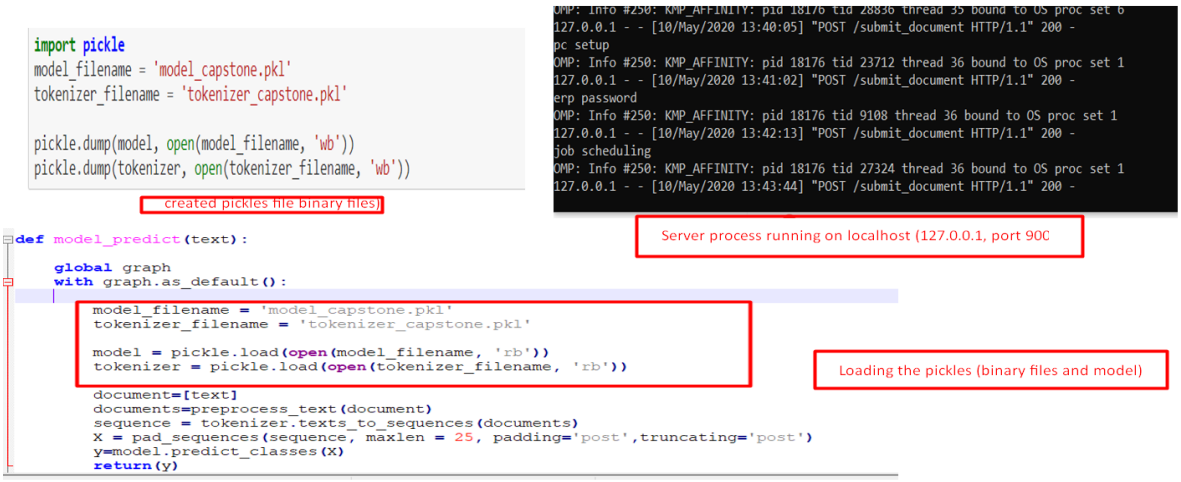
We deployed the model using flask and it is working,

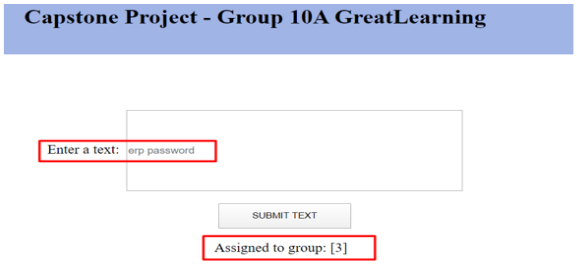
Code is uploaded to git hub repository under ML_APP folder.

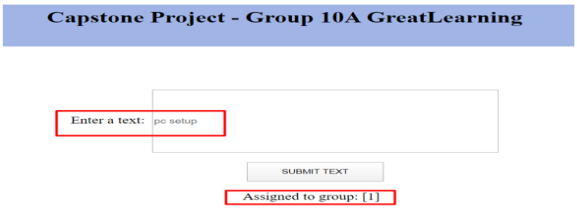
One can download the folder and start a local website with below command

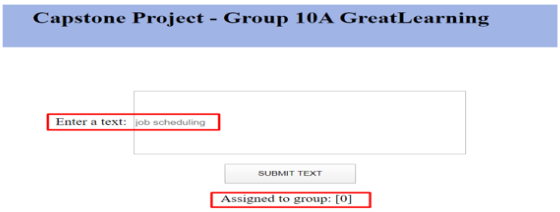
python app.py

https://127.0.0.1:5000 <- to login the site and start predicting groups based on description provided.

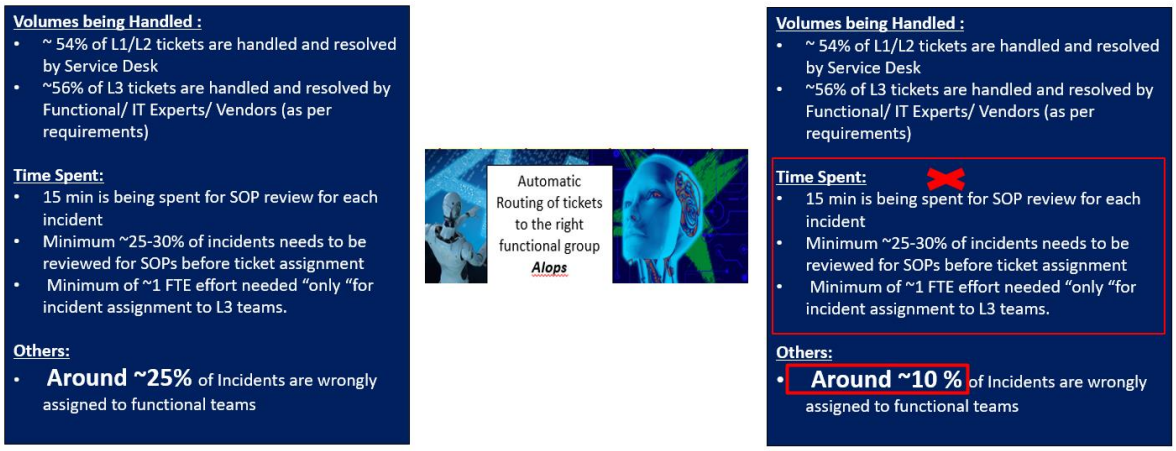








9 Implications



- We can say that our solution an solve then business problem. We can see from the above visualization that the time spent on non-value-added activities are almost eliminated as it can be automated by our Model
- Also, we could reduce the wrong assignment of the tickets to around 10% as compared to 25%
- As the non-value-added activities have been almost eliminated, the resources can be used to solve the ticket rather than doing non-value-added activities. It will improve the user satisfaction

10 Limitations

Our major limitation seems to be the original dataset. Considering the number of numbers of assignment groups and the dataset distribution, it was difficult to keep the same assignment groups and create the model. Even after creating 4 groups the data could have been balanced better.

We also seem to have other language (possibly German) in the dataset but in our model, we used only English dictionary

Also, we used short description text as the text for model building. After text processing, the max length of text was around 25. In real world, the text is likely to be much bigger in length (meaning full text). It will help to really create more meaningful assignment groups

Enhancing the solution:

Possibly, the following should have helped to enhance our solution

- More data sets from each group can help better learning
- Balanced Data Sets
- We could have used foreign dictionaries for better understanding of words
- Use of Fast Text for word embeddings
- This helps capture the meaning of shorter words and allows the embeddings to understand suffixes and prefixes.
- fast Text works well with rare words. So even if a word wasn't seen during training, it can be broken down into n-grams to get its embeddings.
- Word2vec and Glove both fail to provide any vector representation for words that are not in the model dictionary. This is a huge advantage of this method.

• Use of advanced layers (Attention Layers)

Final state of the encoder RNN/ LSTM ultimately decides the decoding process, or at least heavily influences it, while the previous states do not have any influence over the decoding process. Next observation is that the final encoder state

• Attention Mechanism does – it allows the decoder to pay attention to different parts of the source sequence at different decoding steps.

11 Closing Reflections

What we you learned from the process? What we do differently next time?

1. Often users are unaware of the exact problem or do not know what all details might be important for solving the problem and end up not specifying relevant information. For example, in a lot of IT support tickets the name of the operating system, application, version and other important contextual information are omitted. Without this information it may be di cult to drill down to the exact problem category and resolver group. Thus, we should augment the text data with context information and insights obtained from the data to create a better ticket which helps in improving the prediction of resolver group and problem category leading to faster ticket resolution.

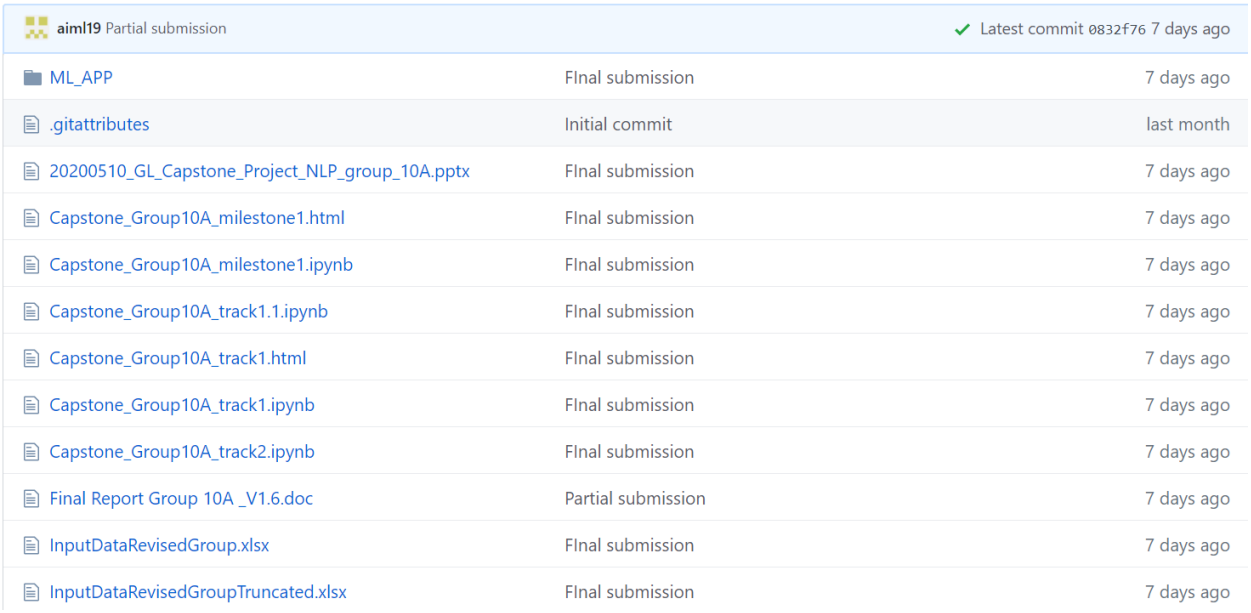
2. Our data contains a lot of non-English comments, a dictionary/NLP process incorporation of non-English words into AIML analysis will strengthen the solution

3. An end-to-end system which can analyze image content in tickets ,understand the nature of the problem indicated in the image and automatically suggest a resolution .Like a specific type of attachment, viz. screenshots, as this is the most common type of attachment, requiring human supervision, found in IT support tickets

12 GitHub repository

GitHub repository with documentation, code and programs

https://github.com/aiml19/aiml



aiml19 Partial submission		Latest commit 0832f76 7 days ago
ML_APP	FInal submission	7 days ago
.gitattributes	Initial commit	last month
20200510_GL_Capstone_Project_NLP_group_10A.pptx	FInal submission	7 days ago
Capstone_Group10A_milestone1.html	FInal submission	7 days ago
Capstone_Group10A_milestone1.ipynb	FInal submission	7 days ago
Capstone_Group10A_track1.1.ipynb	FInal submission	7 days ago
Capstone_Group10A_track1.html	FInal submission	7 days ago
Capstone_Group10A_track1.ipynb	FInal submission	7 days ago
Capstone_Group10A_track2.ipynb	FInal submission	7 days ago
Final Report Group 10A _V1.6.doc	Partial submission	7 days ago
InputDataRevisedGroup.xlsx	FInal submission	7 days ago
InputDataRevisedGroupTruncated.xlsx	FInal submission	7 days ago

13 Reference

https://towardsdatascience.com/topic-modelling-in-python-with-nltk-and-gensim-4ef03213cd21

https://www.machinelearningplus.com/nlp/lemmatization-examples-python/

https://www.youtube.com/watch?v=5BVebXXb2o4

(There are six videos)

https://keras.io/api/preprocessing/

https://www.tutorialspoint.com/keras/keras_layers.htm