AIML Online Capstone -AUTOMATIC TICKET ASSIGNMENT

Submitted By:

Utkarsh Khamgaonkar

Krupa HS

Sai Tejaswi

Manish Raj

Batch:

G3 – NLP A

Jan A – Grp 3 - GL

Jan '20 – Jan '21

INDEX

Topics Page No.

1. Summary of problem statement, data and findings

Problem Statement:

A critical aspect of any IT function is 24/7/365 availability of its applications, also known in Business jargons as "“Keeping the lights on!".

Companies often employ Incident Management processes to address any unplanned interruption to an IT service or reduction in the quality of an IT service that affects the Users and the Business, also known as an "incident”. 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.

The key is to assign the incoming incidents to correct stakeholder team (or Assignment Group) and to do it quickly in order to reduce that overall turnaround time (TAT). This process, if done manually, is prone to errors as well as delays, which can bring down the overall Customer Satisfaction.

In this capstone project, the goal is to automate the Incident Assignment process by building a classifier that can classify the tickets by analysing text using NLP libraries.

Data Provided:

A snapshot of the provided dataset.

	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

Here the data provided consists of below 4 columns:

1. Short Description: A brief introduction of the reported issue.
2. Description: Details of the reported issue.
3. Caller: One who has reported the issue.
4. Assignment Group: The team which is responsible for providing resolution of issue to the caller.

Initial Findings:

We observe that the columns 'Short Description' and 'Description' are of vital importance to draw a pattern in the incidents and related them to a particular 'Assignment Group' which is our target variable here. Since, the column 'Caller' does not contribute in the classification of tickets, it can be safely deleted during EDA before moving onto Text Pre-processing stage.

We'll start out by importing the libraries essential for executing this project.

2. Overview of the final process

The problem at hand is a classic case of multi-
class classification, where based on the short description and description, we have to c lassify a ticket in one of the many available assignment groups.

In order to reach upto the level of using the given data to train various classification models (both machine learning models as well as neural networks), it is imperative th at we process and clean the data.

We started by checking for and treating if found, the null values and duplicate rows in the given dataset. Then we dropped the 'Caller' column since it was not important fo r the final classification process.

Then we started doing the text preprocessing by first fixing the weird characters (enco ding issue) in the given dataset. Then we cleaned the text using RegEx patterns and str ip off the punctuations and non-
important parts of text like numbers and email address. At this point, we found some more duplicates which we removed while keeping just one copy.

Then we translated all the non-
English text into English, so that we can proceed with lemmatization and text visualiza tion like WordCloud and WordFrequency. Then we combined the two text columns, i.e., 'short description' and 'description' into one (to be used as an input column for m odel training later), and converted the 'assignment groups' columns into a categorical column (to be used as a output column for model training later). Then we divided the input and output columns in the ratio of 70:30 for train and test sets.

We used below algorithms (models) in our solution for this problem statement.

1. Multinomial Naive Bayes
2. k-Nearest Neighbour
3. Linear SVM
4. Decision Tree
5. Random Forest
6. Logistic Regression
7. AdaBoost

8. Gradient Boosting
9. XGBoost
10. Bagging
11. Stochastic Gradient Descent
12. Neural Network
13. Simple LSTM
14. Gated Recurrent Unit
15. Recurrent Convolutional Neural Networks
16. Bi-Directional LSTM

Based on the accuracy that we got while training all these models, we also incorporat e various techniques like resampling of data, using callbacks like Early Stopping etc.

3. Step-by-step walk through the solution

We started off by studing the given dataset. At first glance we could see that there we re records where the text (short description and/or description) was having weird char acters instead of legitimate language text. This information prompted us to analyse ho w to fix this issue using **ftfy (fixes text for you)** library.

After fixing the encoding issue, the next thing that we observed was the fact there wer e many instances where a particular caller had raised multiple tickets.
This hinted on the possibility of having duplicate records of same issue getting recorde d by multiple users. This led us to cleaning the text off punctuations, number and ema il IDs, so that we could finally find duplicate rows in the data and remove them befor e proceeding further.

Once we had removed the duplicate rows, the next step that we felt was logical and i n-
line with our approach so far was to get all the text converted into English. This not o nly enabled us to use just English based lemmatizer (**spaCy**), but also helped us in getti ng ready for model training.

Then we combined all the text (which is lemmatised by now, so it is just words anyw ays!) into one column so as to simplify model training process by keeping all input in a single place. Also, we converted the 'assignment group' column which is given as stri ng into numerical categories, which would help in the final classification process.

Then we started training all the above mentioned machine learning models one by on e, only to see that while we are getting better (sometimes marginally, sometimes rema rkably!) training accuracy, the testing accuracy was generally on the lower side, hintin g on the overfitting of data, which can be caused if there is any imbalance in the data favoring one particular assignment group over others. This led us to visualise the assig nment group data in bar chart and pie chart format to confirm that assignment group 0 tickets covered nearly half of the provided data set. Also, there were many groups whose ticket counts were so low that clubbing such groups into one bigger group wo uld have further simplified our solution. After clubbing the minor groups into one big group and treating data imbalance by using resampling, our machine learning models started showing desired level of accuracy.

For neural networks, we started off with a simple deep neural network, which gave a poor accuracy score of 48%. To improve the accuracy, we included techniques like Ea rly Stopping callback, use of LSTM/GRU/RNN/Bi-
directional LSTM, however the accuracy improved only marginally, leading us to our conclusion that for this problem statement, machine learning models with resampled data are suitable for classifying any ticket that IT team may get.

4. Model evaluation

Details of final model:

RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,criterio n='gini', max_depth=None, max_features='auto',max_leaf_nodes=None, max_sampl es=None,min_impurity_decrease=0.0, min_impurity_split=None,min_samples_leaf= 1, min_samples_split=2,min_weight_fraction_leaf=0.0, n_estimators=100,n_jobs=No ne, oob_score=False, random_state=None,verbose=0, warm_start=False)

Objective of evaluation: To get the best performing models out of all the models trai ned, so as to get the best accuracy on real-time ticket data.

Prominent parameters:

1. criterion='gini'
2. min_samples_leaf=1
3. min_samples_split=2
4. n_estimators=100

We evaluated the success of our models by comparing their accuracy over test data an d found that Random Forest classifier performed the best with oversampled data.

5. Comparison to benchmark

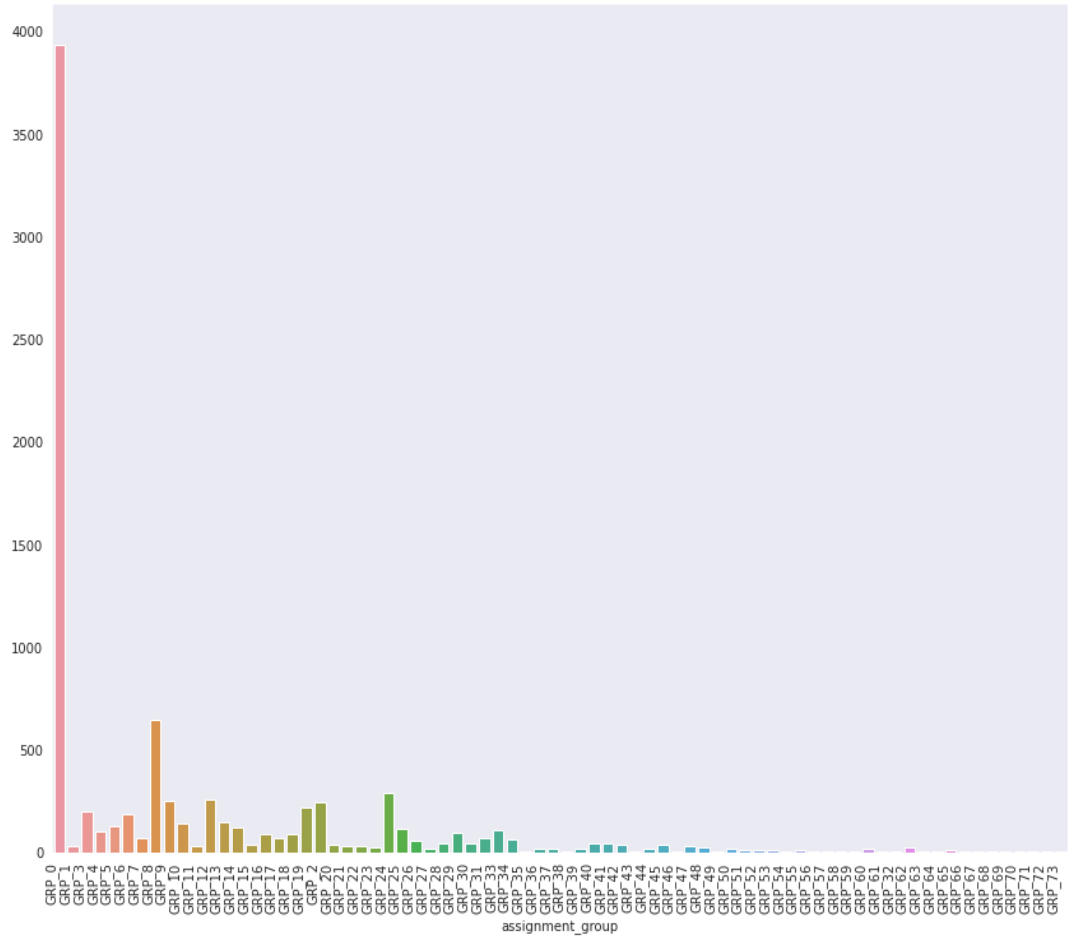
At the outset, keeping in mind the size of the given dataset, challenges like encoding is sues present in the text, time out issues while trying to translate the text into English, a nd time-
taking I/O operations like unzipping and reading of GloVe file for word embedding, we kept our benchmark at ~95% accuracy.

With the help of techniques like resampling of imbalanced data, and use of proper par ameters, our final model (**Random Forest Classifier**) was able to perform at a test accu racy of 99%, thereby improving on the benchmark.

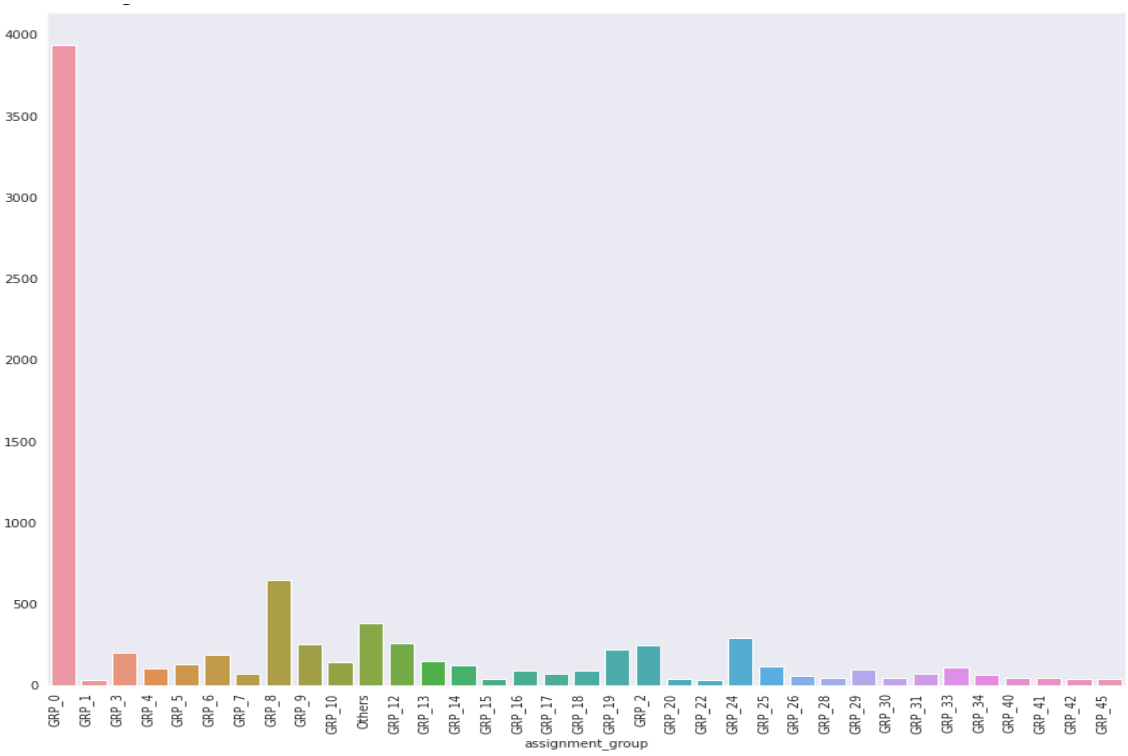
6. Visualizations

Some of the visualizations that we used in our project is as follows.

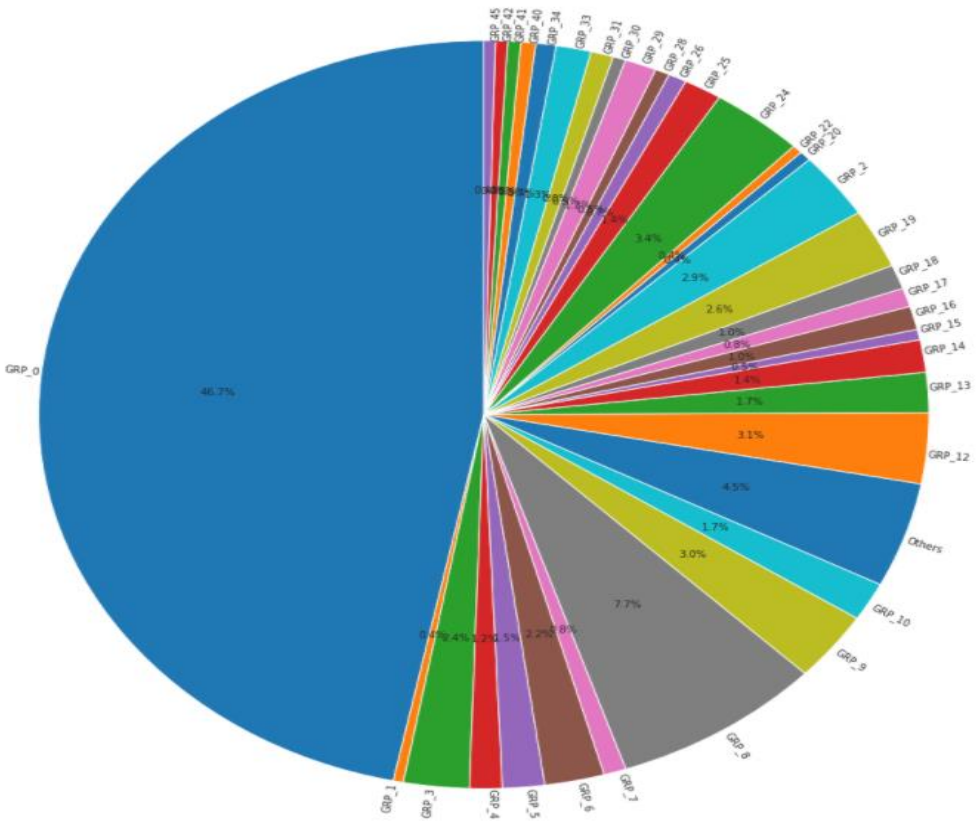
i. Distribution of data across originally given 74 assignment groups.



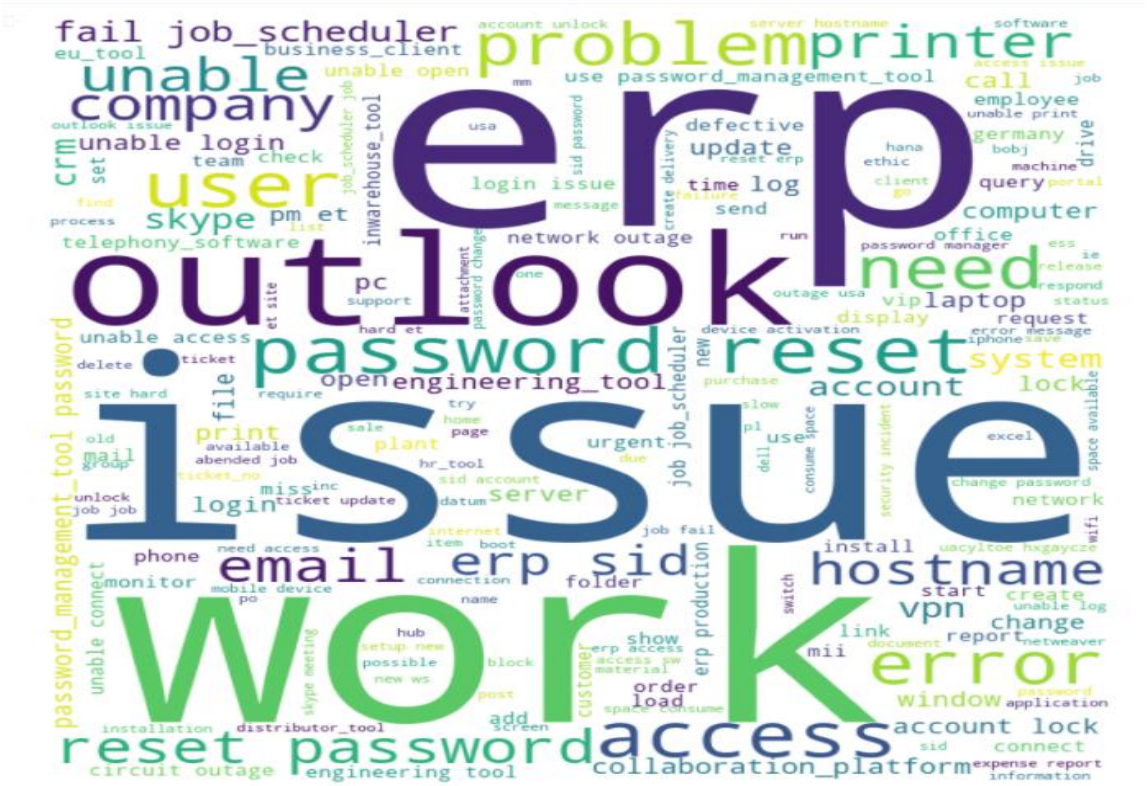
ii. Distribution of data across reduced 35 assignment groups.



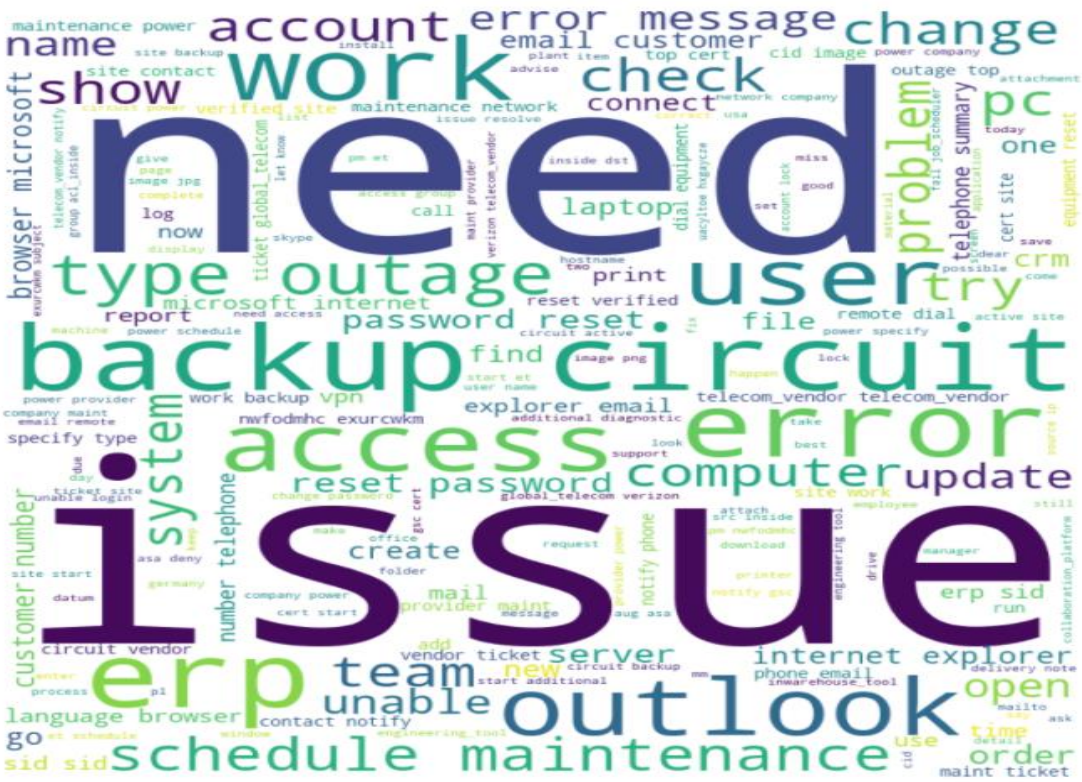
iii. Pie chart to show data distribution on reduced number of assignment groups.



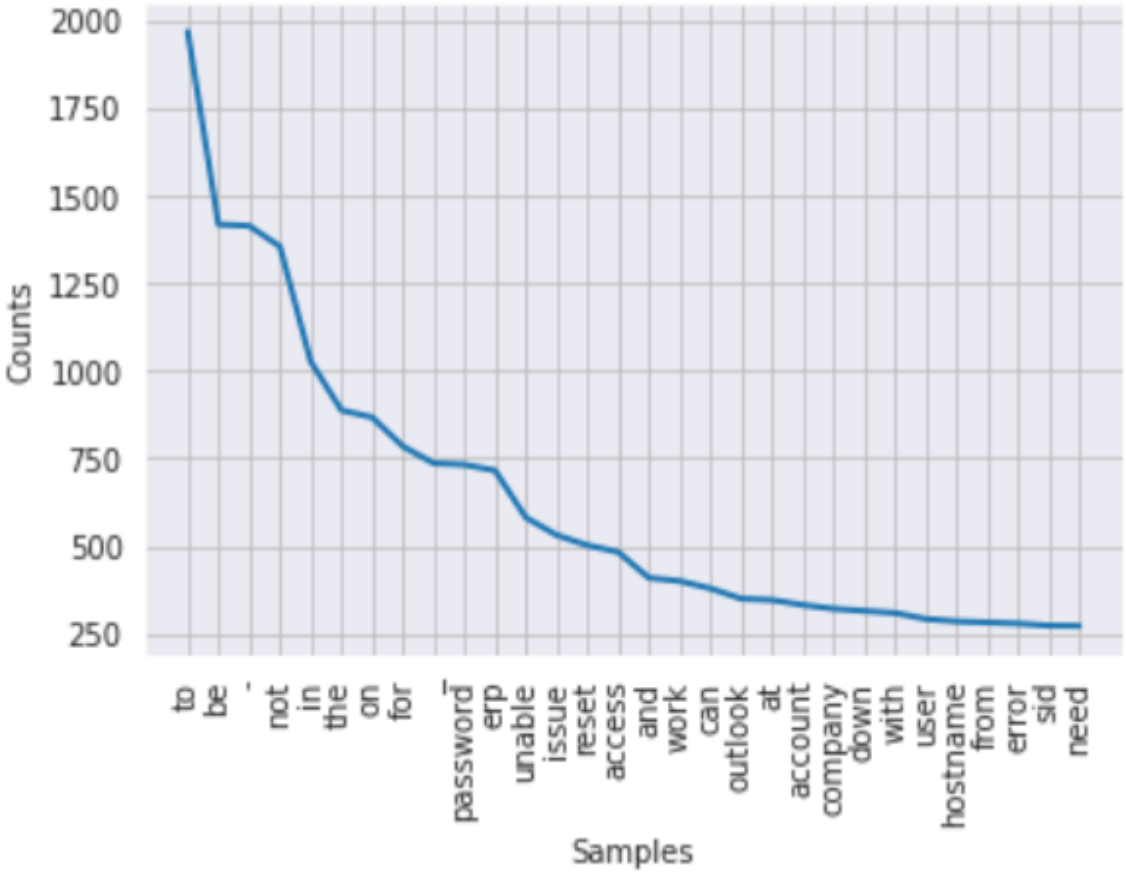
iv. Word cloud for ‘short description’ column



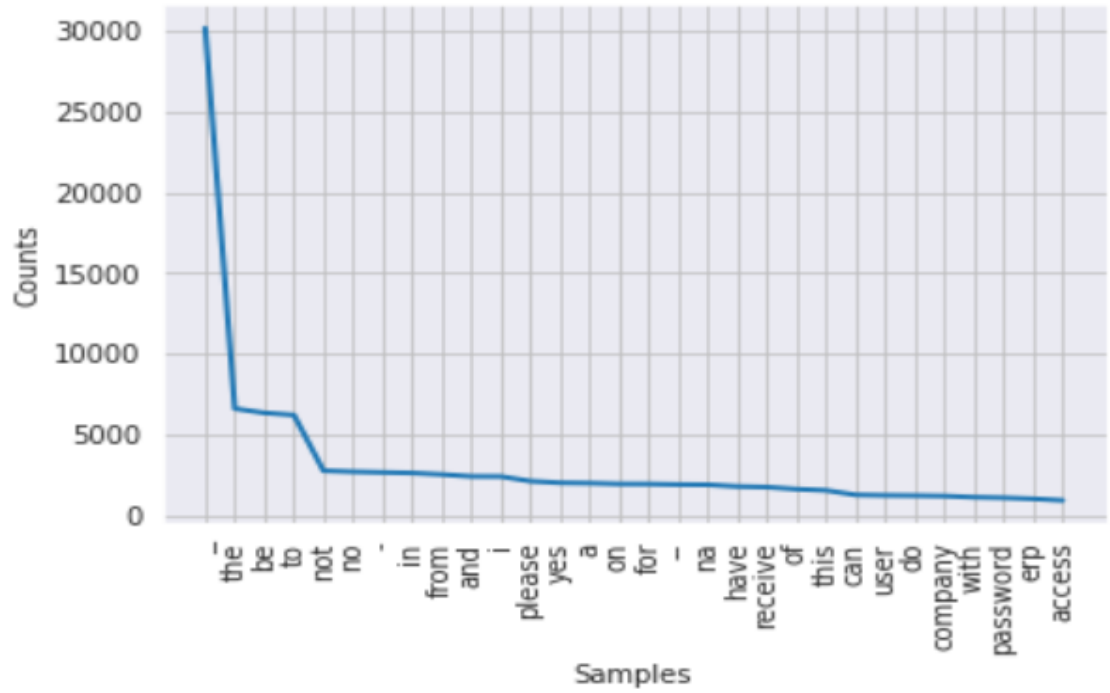
v. Word cloud for ‘description’ column



vi. Word frequency for 'short description' column



vii. Word frequency for 'description' column



7. Implications

With both training and testing accuracies of over 99%, confusion matrix showing minimum False Positives or False Negatives, Precision, Recall and F1 Scores close to 100% mark, we believe this solution addresses the problem statement correctly and completely.

With a very high level of confidence, we can recommend this solution to be used for real-time ticketing data.

8. Limitations

This solution may underperform if the size of data is very large (may be around 1,00,000 records). In such scenarios, we will have to look for better performing but complex RNN based architectures like BERT which is highly optimised and capable of handling very huge volumes of data.

9. Closing Reflections

Lessons learnt from this Capstone project are as follows:

i. Always keep checkpoints in a big project like this so that work can be resumed from the point it was left and the earlier cells are not needed to be executed again, thus saving a lot of time.

ii. In scenarios where we are using certain packages which make http calls to external APIs, effective usage of **time.sleep()** can help us avoid 'Too Many Requests' exception.

iii. Proper use of functions can avoid a lot of code duplication, thereby enhancing the readability of the code.