

## AUTOMATIC TICKET ASSIGNMENT

**GROUP 3 (Capstone Group 3\_NLP June)**

1. **Siddhartha Borpuzari**
2. **Sarat Kumar Behera**
3. **Chhaya Jhanwar**
4. **Prashant M**
5. **Premalatha K**

**INDEX**

Summary of problem statement, data and findings 3

Overview of the final process 4

Step-by-step walk through the solution 8

Model Evaluation 13

Comparison to benchmark 17

Implications & Future scope 17

Limitations 18

Reference 19

**Summary of problem statement, data and findings:**

In today’s world most of the businesses are digitized and any disturbance in digital space will cause manpower loss, productivity loss which can lead to monetary loss to company and cause intangible loss to company’s brand. To ensure continuity of business operation, IT functions play important role by keeping all digital assets and its related services running to its optimum working condition. To meet this objective IT leverages Incident Management process. Incident is raised when there is unplanned interruption in IT services or reduction in the quality of an IT service that affects user and business. Using Incident management process. IT provides solutions or quick fix /workarounds to resolve reported interruption and restores the service to its full capacity. In most of the organizations incidents are raised by its various Business and IT users, End users and some time by its vendors using integrated monitoring systems and tools.

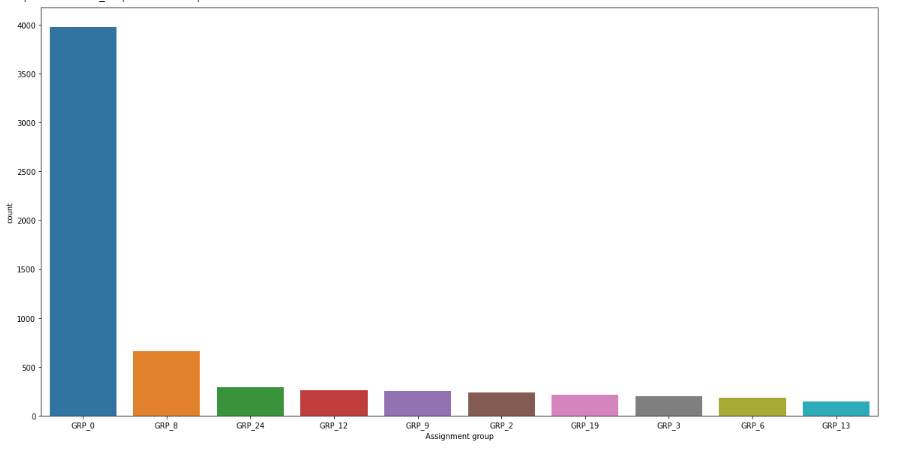
The current incidents resolution consists of assigning tickets to L1/L2 service deck teams which normally resolves 54% raised tickets by proper categorization and carrying out initial diagnosis. The tickets which are not able to resolve by L1/L2 team then will escalate to L3 team. Normally 46% of tickets are resolved by L3 team. L1 / L2 needs to spend time reviewing Standard Operating Procedures (SOPs) before assigning to Functional teams (Minimum ~25-30% of incidents needs to be reviewed for SOPs before ticket assignment). 15 min is being spent for SOP review for each incident. Minimum of ~1 FTE effort needed only for incident assignment to L3 teams. During the process of incident assignments by L1 / L2 teams to functional groups, there were multiple instances of incidents getting assigned to wrong functional groups.

This project is an attempt at Leveraging Machine Learning and Artificial intelligence to automatically classify tickets. By assigning the tickets to the right owner in a timely manner the organization saves effort, increase user satisfaction and improve throughput in the ticketing pipeline.

The data includes 4 columns including short description, Description, caller and assignment group. The caller group includes the names/details of the caller but does not have any direct influence on the data. The assignment group is decided on the basis of the judgement of the person handling the call. However the identity of the caller does not have a much important role to play and hence caller column can be omitted.

The short description includes a brief mention of the issue and it has more details in the description column where other details like the caller name or what was the output on screen that the user received.

The data consists of 8500 rows and 4 columns and the columns include null values as well. There are 74 unique groups to which the tickets are assigned and mostly to GRP\_0 (46.8%), followed by GRP\_8, GRP\_24 and so on.



* The objective of this capstone project is to build an automated classifier system using Machine learning and Artificial intelligence technique that can classify the tickets and assign it to the right functional group by analyzing user entered text.
* The business objective is to reduce the manual effort thereby improving the productivity.

**Overview of the final process:**

**Handling the Missing values in the data set:**

**Description column:**

There is 1 row with null value in description column. The topmost issue is repeated in the column is "the" with 57 occurrences. There are 7817 unique words. From the Word Cloud, the topmost text is "received", "Gmail", "unable", "access" etc..

**Short Description column:**

There are 8 Missing values in “Short Description” column. The topmost issue is the repeated in the column is "password reset" with 38 occurrences. There are 7482 unique words. From the Word Cloud, the topmost text is "issue", "outlook", "can’t", "login" etc

Replaced the “Short Description with “Description”. This will help us not to loss any data and also with replacement of right text of problem.

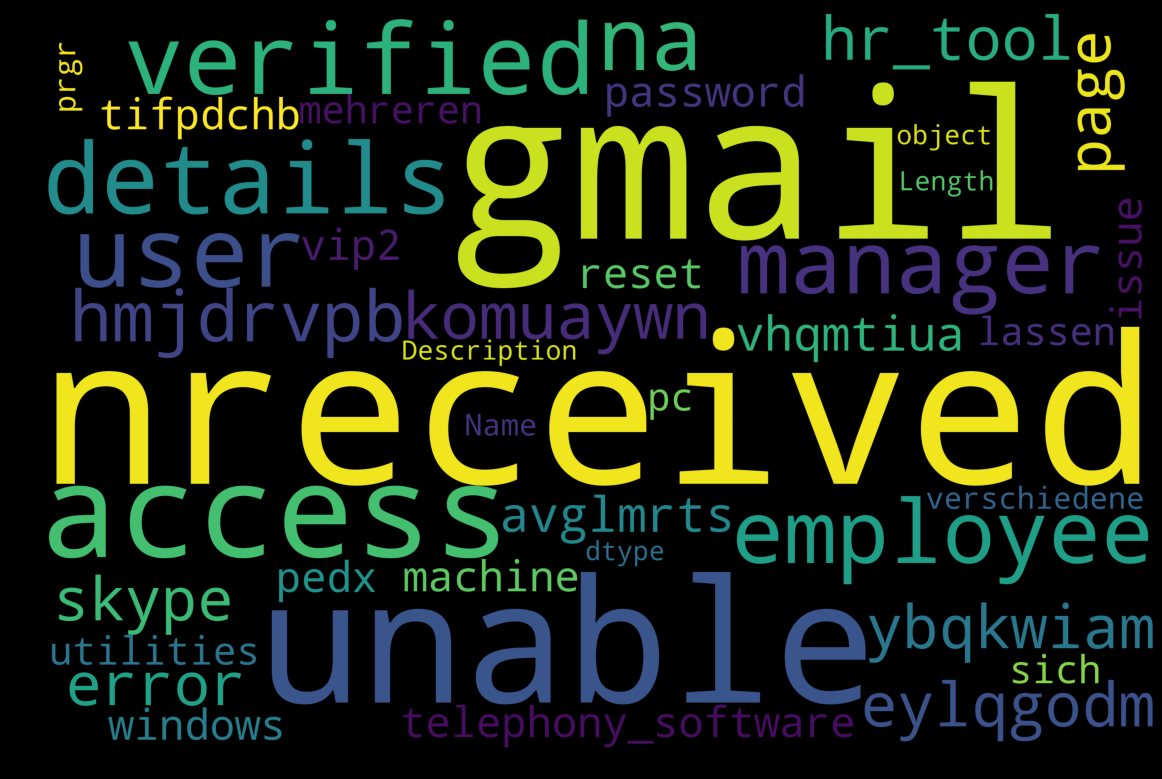


Fig : Most common words in “Description” column before data pre-processing

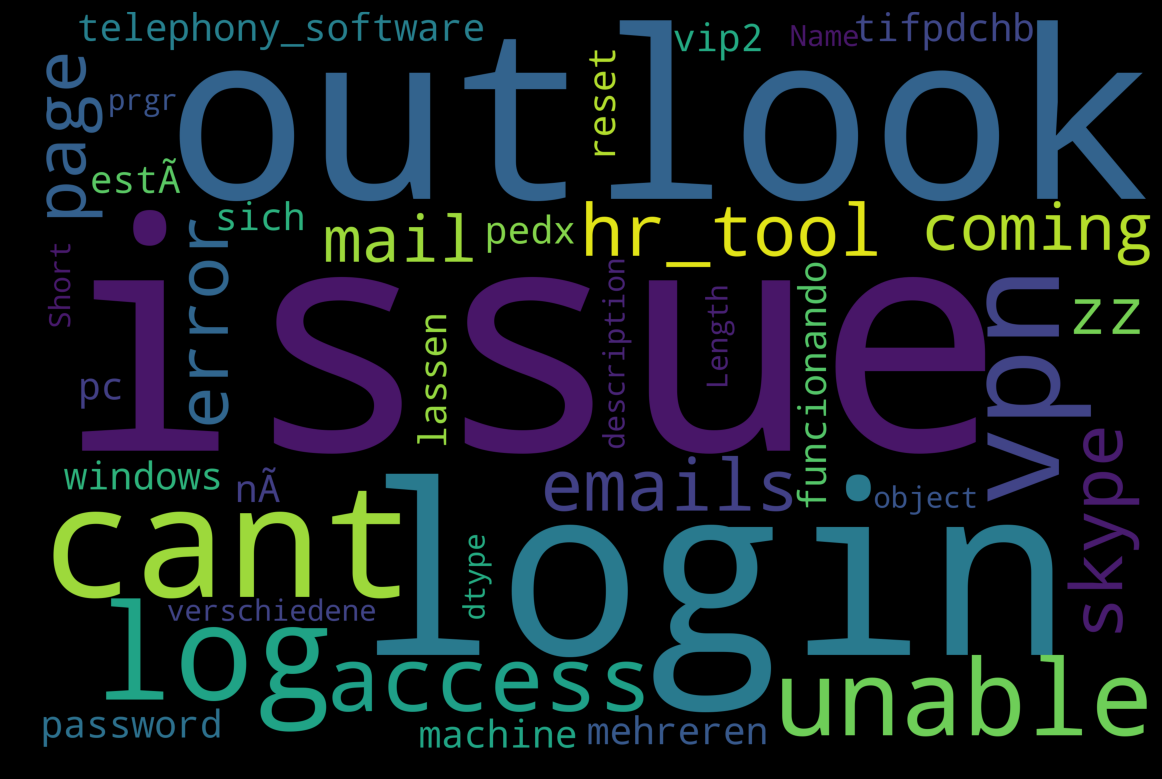


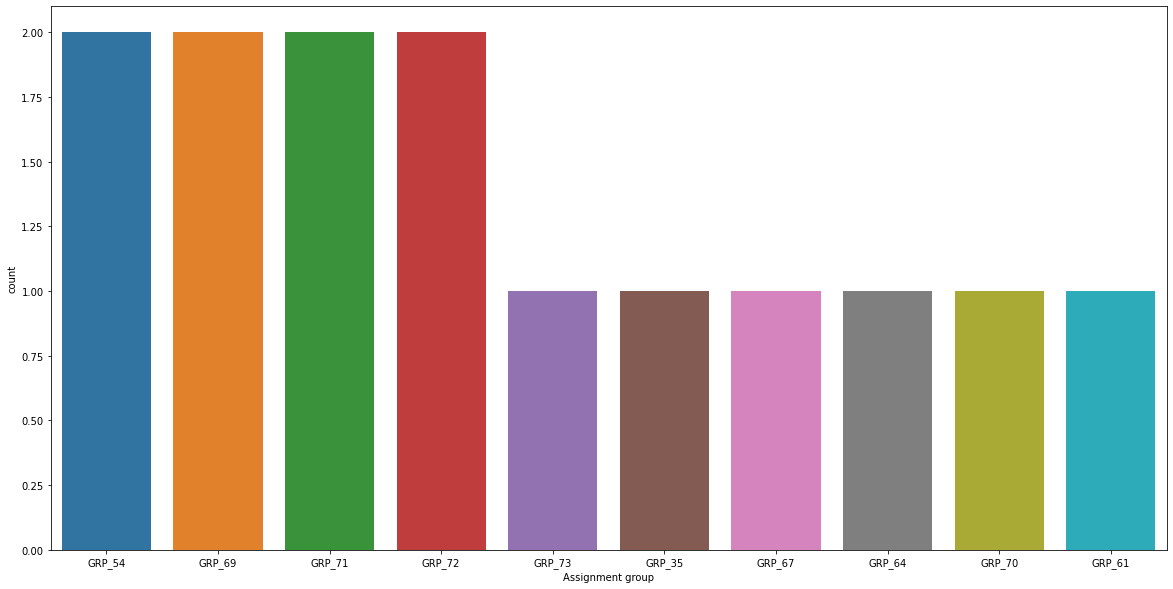
Fig : Most common words in “short description” column before data pre-processing

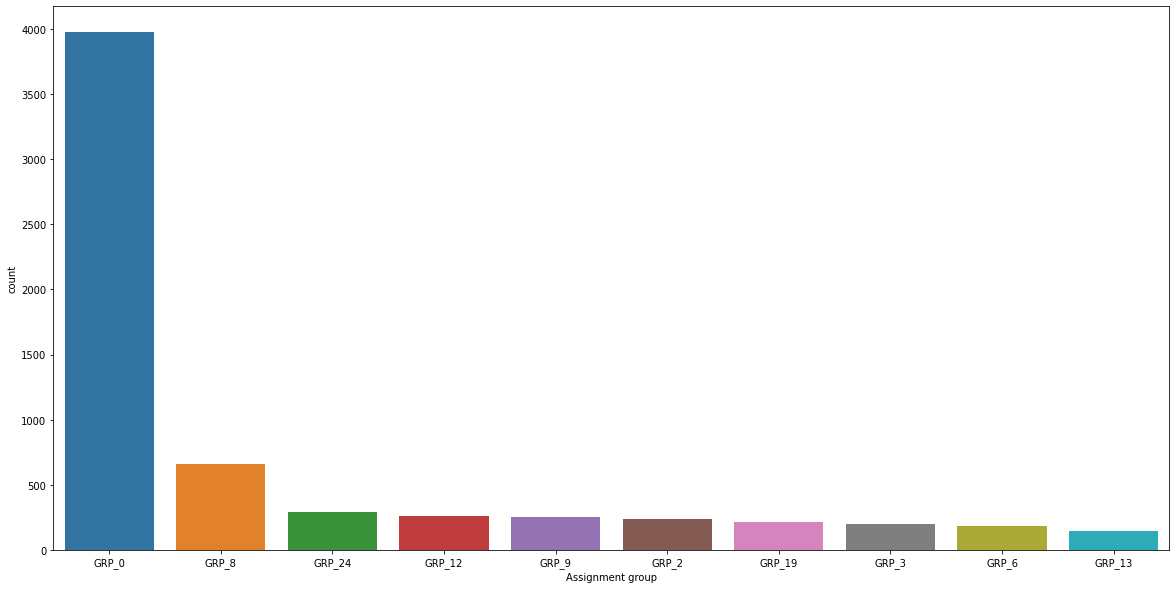
**Caller column:**

The “caller” column doesn’t have any direct influence on the output and hence we can drop the Column.

**The Assignment Group Column:**

There are no null values. We see that there are total 74 assignments groups in the data. Top 1 group worked on 46% of the tickets. Top 1-Top 3 worked on 57% of the tickets. Top 1-Top 5 worked on 64% of the tickets. Top 1-Top 10 worked on 75% of the tickets. Bottom 10 groups contain 1 or 2 cases assigned to them.



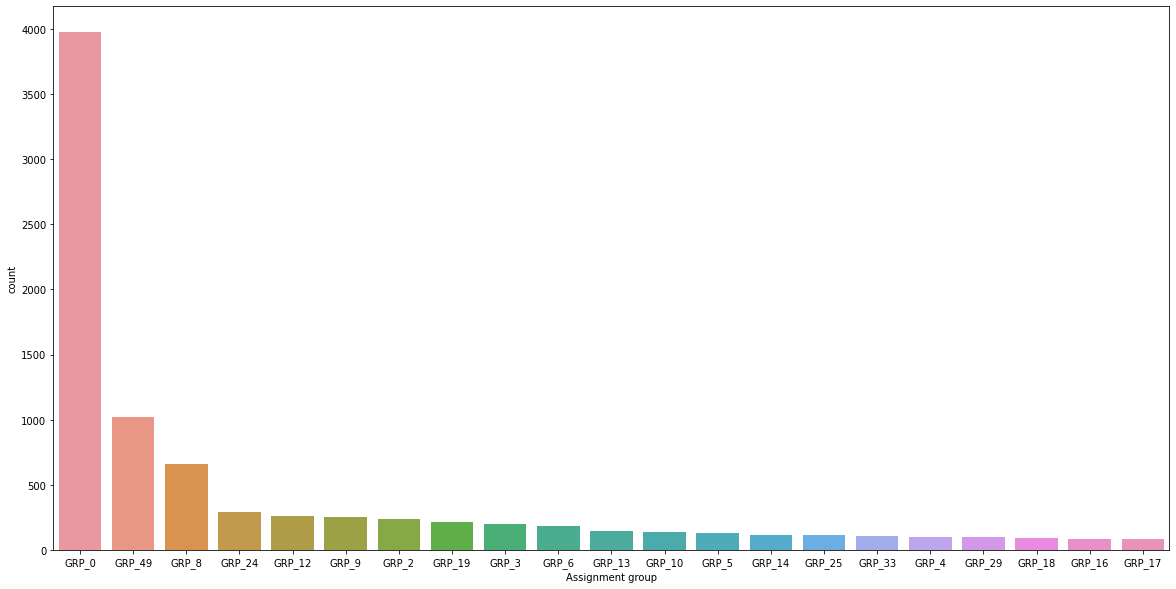


**Text Pre-processing:**

Text pre-processing is the process of transferring text from human language to machine-readable format for further processing. After a text is obtained, we start with text normalization. Text normalization includes: -Converting the all texts into one language i.e. in English

* + - Translate all text into one language as English.
    - converting all etters to lower or upper case
    - removing punctuations, accent marks and other diacritics
    - converting numbers into words or removing numbers
    - removing white spaces
    - removing stop words, sparse terms, and particular words
    - text canonicalization
    - As the texts in “description” and “short-description” are similar they can be merged to reduce redundancy of data.

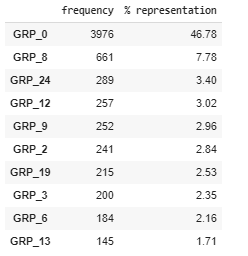
Using Fuzzywuzzy library the similarity ratio between the texts of the imbalanced groups are found out. It has been found that “GRP\_49” showed most similarity to the imbalance groups with a similarity of 75% and above.Number of groups having data coverage greater than 1% are 20 and No of groups that are data coverage of less than 1% are 54. So we have converted the less than 1% into one Group now the imbalanced groups are attributed to GRP\_49 thereby making the total no of target groups to 21.



**Handling imbalanced data:**

The “Assignment group” data is highly imbalanced with nearly 47% of the data belonging to the class “GRP\_0”. Out of 74 unique assignment groups only 20 groups had presence in more than 1% of the data. So the rest 54 groups could be attributed to the group with most similar text in the final description column.

The frequency of the most occurring classes are as follows :



The 20 unique groups each representing more than 1% of the data are named as balanced and the rest 54 groups are named as imbalanced.

Using Fuzzywuzzy library the similarity ratio between the texts of the imbalanced groups are found out.

It has been found that “GRP\_49” showed most similarity to the imbalance groups with a similarity of 75% and above.

The imbalanced groups are attributed to GRP\_49 thereby making the total no of target groups to 21.

A new dataset is carved out without any null value having 1 text column and 1 target column with 21 target groups.

**Step-by-step walk through the solution:**

**Vectorizing the texts**

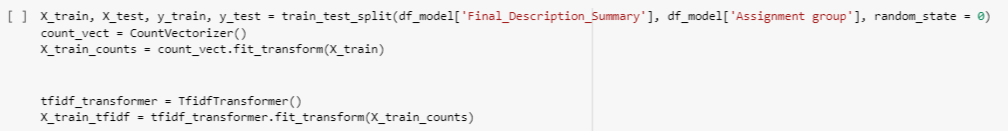
The texts of the final data is lemmatized and the words are converted to its base form. We used Spacy’s lemmatizer to obtain the base form of the words.



The raw data, a sequence of symbols cannot be fed directly to the algorithms themselves as most of them expect numerical feature vectors with a fixed size rather than the raw text documents with variable length. Before creating the above classifier models, let's first vectorize our inpur data. Scikit-learn's CountVectorizer can be used to transform a corpora of text to a vector of term / token counts. It also provides the capability to preprocess your text data prior to generating the vector representation making it a highly flexible feature representation module for text.

The data is divided into training and testing at 75% ratio with the column “Assignment group” being labelled as target group(Y) and the “Final\_Description\_summary” column as independent value(X).

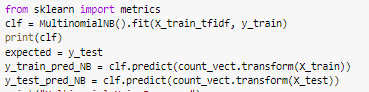
The text is vectorized using CountVectorizer and then Tf-IDF transformation is applied. TF-IDF algorithm helps sort data into categories, as well as extract keywords.



**Application of Machine learning models**

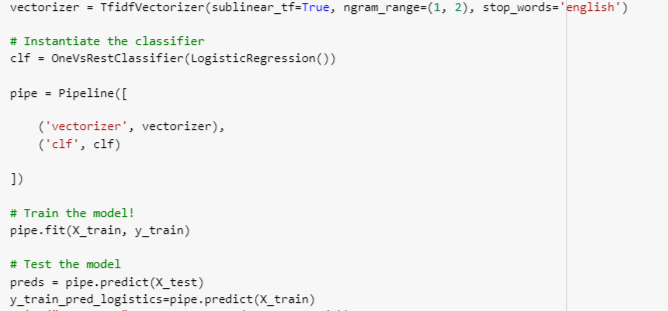
**Multinomial Naïve-Bayes:**

In this algorithm there is a strong assumption that every feature is independent of the others, in order to predict the category of a given sample. They are probabilistic classifiers, therefore will calculate the probability of each category using Bayes theorem, and the category with the highest probability will be output.



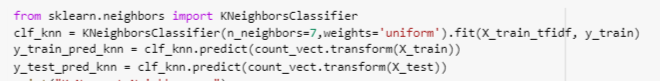
**OneVsRest Classifier:**

Logistic regressions are only binary classifiers and cannot handle target vectors with more than two classes. However, there are extensions to logistic regression to do that. In one-vs-rest logistic regression (OVR) a separate model is trained for each class predicted whether an observation is that class or not (thus making it a binary classification problem). It assumes that each classification problem (e.g. class 0 or not) is independent.



**K Nearest Neighbor**:

The k-nearest neighbor (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.KNN requires scaling of data because KNN uses the Euclidean distance between two data points to find nearest neighbors. Euclidean distance is sensitive to magnitudes. The features with high magnitudes will weight more than features with low magnitudes.



**Support Vector Classifier (SVC):**

Support vector machines are a set of supervised learning method used for [classification](https://scikit-learn.org/stable/modules/svm.html#svm-classification), [regression](https://scikit-learn.org/stable/modules/svm.html#svm-regression) and [outliers detection](https://scikit-learn.org/stable/modules/svm.html#svm-outlier-detection).Being a classification problem at hand we are using the support vector classifier.

The advantages of support vector classifiers are:

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.



**Decision Tree**:

It is a non-parametric supervised algorithm used for [classification](https://scikit-learn.org/stable/modules/tree.html#tree-classification) and [regression](https://scikit-learn.org/stable/modules/tree.html#tree-regression). The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Certain benefits include:

* Requires little data preparation. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
* The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
* Able to handle both numerical and categorical data. Other techniques are usually specialised in analysing datasets that have only one type of variable.
* Able to handle multi-output problems.

Decision tree is sensitive to noisy data. It can overfit noisy data. The small variation in data can result in different decision trees. This can be reduced by bagging and boosting algorithms.



**Random Forest Classifier**:

Random forest consists of a large number of individual decision trees that operate as an [ensemble](https://en.wikipedia.org/wiki/Ensemble_learning). Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes the model’s final prediction.

Uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The various decision trees in the random forest model protect each other from their individual errors (as long as they don’t constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction

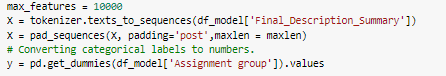
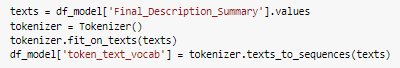
In this model we have taken 1000 decision trees with maximum no of features as 20 . The overall accuracy turns out to be 65% which is quite improvement over the individual decision tree performance.



**Application of LSTM model**

LSTM is a special type of Recurrent neural network that preserves long term dependency in a more effective way compared to the basic RNNs. This is particularly useful to overcome vanishing gradient problem as LSTM uses multiple gates to carefully regulate the amount of information that will be allowed into each node state.LSTM in its core, preserves information from inputs that has already passed through it using the hidden state.Unidirectional LSTM only preserves information of the past because the only inputs it has seen are from the past.

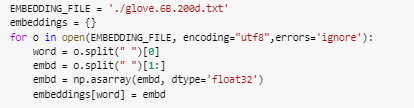
For running this model we retake the original description texts from the final pre-processed data and then tokenize it. Then it is converted to sequences using texts\_to\_sequences command.

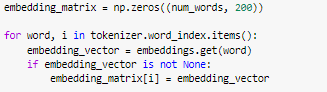


**Embeddings using the pre-trained model glove**

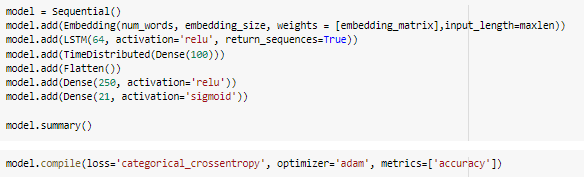
Glove is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Here we are using 'glove.6B.200d.txt' file which is trained on a corpus of 6 billion tokens and contains a vocabulary of 400 thousand tokens.





**Creation of LSTM Model**



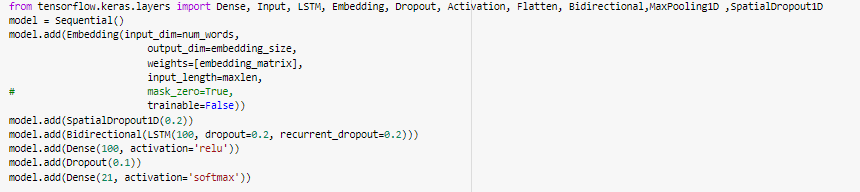
The model was run with a batch size of 100 and over 5 epochs after train and test data split.



**Creation of Bi-Directional LSTM Model**

Using bidirectional will run the inputs in two ways, one from past to future and one from future to past.

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.



The model was run with a batch size of 32 and over 10 epochs.



**Model Evaluation:**

**Comparison of models**

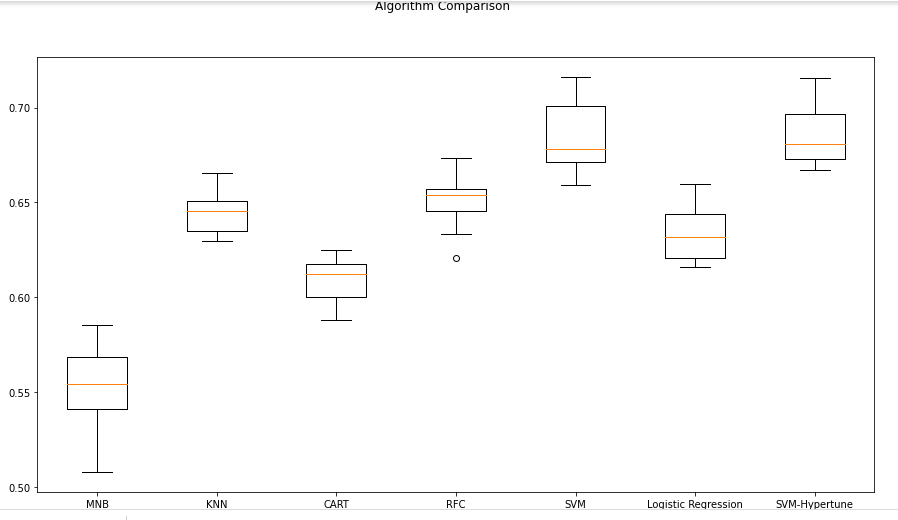
The final outputs obtained from the various models are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Train accuracy (%)** | **Test accuracy (%)** | **ROC\_AUC\_Score** | **F1\_score** |
| Naïve Bayes | 60.61 | 61.41 | 55.44 | 61.41 |
| OneVsRest Clf | 68.50 | 65.46 | 62.38 | 65.46 |
| KNN | 67.69 | 65.27 | 64.18 | 65.27 |
| SVC | 88.93 | 69.04 | 64.18 | 69.04 |
| Decision Tree | 65.71 | 54.49 | 63.55 | 54.49 |
| Random Forest | 85.30 | 65.60 | 62.64 | 65.60 |
| LSTM | 46.93 | 45.88 |  |  |
| LSTM(Bi-directional) | 46.68 | 47.18 |  |  |

**Cross Validation**

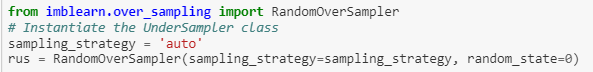
The various machine learning models are re-run with 10 splits of data. The model   
“Support Vector Classifier” is hyper tuned (with parameter C=10,kernel=’rbf’ and gamma=0.1)

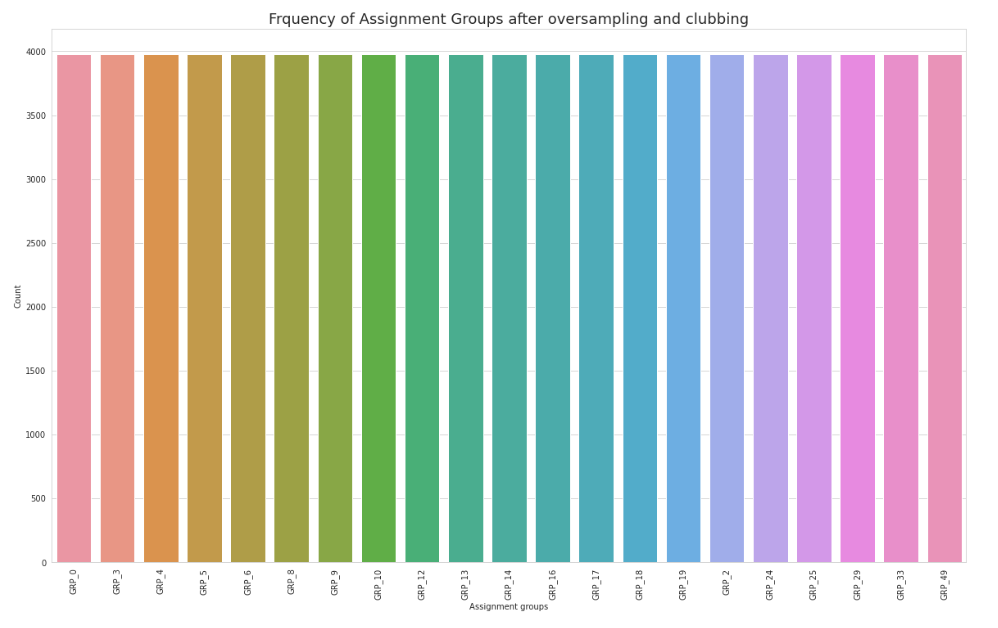
A box plot of the mean of the cross validation scores is illustrated below:



**MILESTONE 2 :**

The data is up-sampled to ensure all 21 groups have equal representation, The machine learning models are fine-tuned to ensure that most optimized parameters are selected.





|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Train accuracy (%)** | **Test accuracy (%)** | **ROC\_AUC\_Score** | **F1\_score** |
| Naïve Bayes(NB) | 82.96 | 82.32 | 90.68 | 82.32 |
| NB -Fine Tuned | 88.89 | 88.25 | 93.78 | 88.25 |
| OneVsRest Clf | 92.25 | 91.51 | 95.52 | 91.51 |
| KNN | 89.32 | 88.45 | 93.93 | 88.45 |
| KNN-Finetuned | 89.32 | 88.45 | 93.93 | 88.45 |
| SVC | 86.97 | 86.27 | 92.77 | 86.27 |
| Decision Tree(DT) | 53.20 | 52.79 | 75.19 | 52.79 |
| DT-Fine Tuned | 47.31 | 46.91 | 72.14 | 46.91 |
| Random Forest(RF) | 91.04 | 90.77 | 95.14 | 90.77 |
| RF-Hypertuned | 88.92 | 88.31 | 93.85 | 88.31 |
| Bagging (DT) | 62.77 | 62.00 | 80.09 | 62.00 |

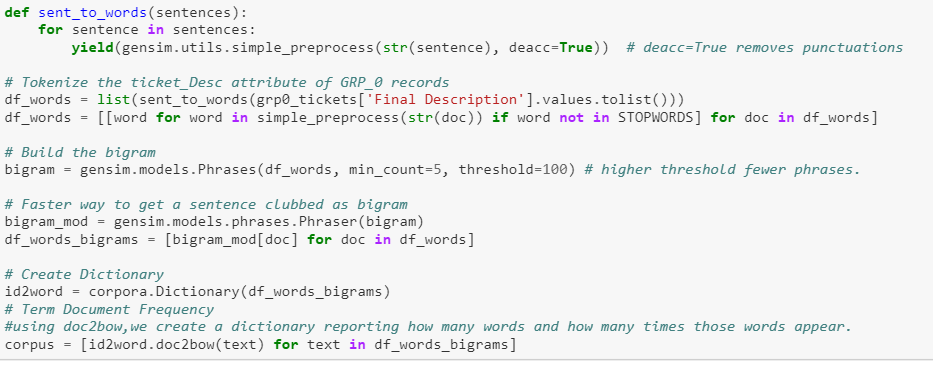
The deep learning models with 12 epochs give accuracy as follows:

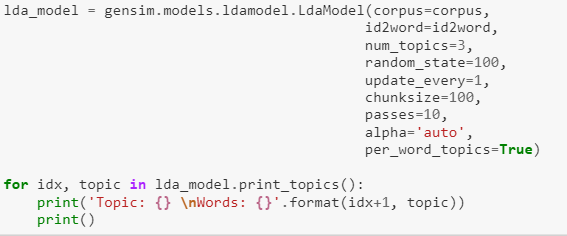
|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Train accuracy (%)** | **Test accuracy (%)** |
| LSTM(Bi-Directional) | 89.71 | 88.94 |
| LSTM | 93.23 | 92.27 |
| GRU | 93.31 | 92.30 |

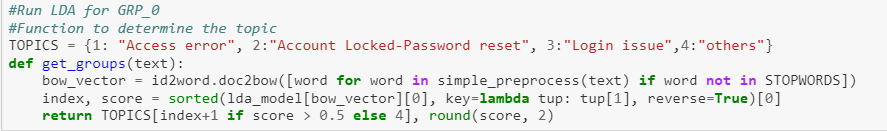
**MILESTONE-3**

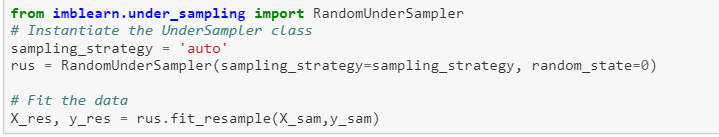
The data can be down sampled with the representation of GRP\_0 being lowered.

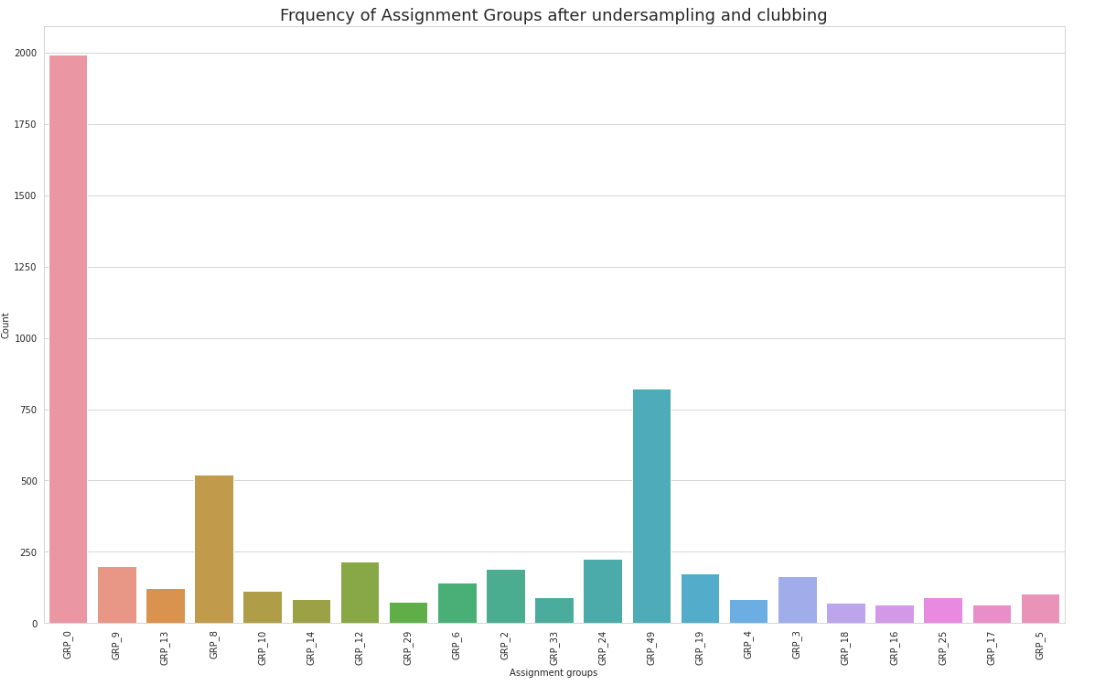
* Used only train dataset for Down-sampling GRP\_0. We are keeping test data as it is , in this case.
* Built a Topic Model with top 3 different topics where each topic is a combination of keywords and each keyword contributes a certain weightage to the topic.
* Ran Latent Dirichlet Allocation (LDA) for each record of GRP\_0 to find the associated topic based on the LDA score.
* As the topic modelling has been trained to accommodate only top 3 or 4 topics for entire GRP\_0 data, any record scoring less than 50%, we categorize them into next (other) topic and such tickets are not the candidates for resampling.
* used RandomUnderSampler, for Down-Sampling.











The dataset when used with LSTM gives an overall accuracy of 35.46%.

**Comparison to benchmark:**

The Best Machine Learning algorithm for this problem is:-

* + OneVesRestLogistic (Train and Test 92% Accuracy using Up-Sampling)

The Best deep learning algorithm is for this problem is:-

* + GRU (Train and Test 93% Accuracy using Up-Sampling)

Use of deep learning method gives train and test accuracy of higher order and hence up-sampling can be considered as a suitable method while handling such imbalanced dataset for text processing.

**Implications & Future scope**

* The scope of the selected model is to assign tickets to the right group for issue resolution. As observed, most of the tickets are related to Password reset, Account lock, Unable to log in for which resolutions are easily available. **So, we can extend this project to provide resolution to commonly occurring issues and only sending non common tickets to the next level where expert involvement is required.**
* We can also extend scope of this project by **implementing an Automated Question Answering model using BOT technique**, where users can type issues in the question field and bot provide answers in form of resolution by running a selected AI model in background.
* **Urgency classification**: We can extend the scope to categories issues in terms of their urgency. If text contains words such as ‘right away’, ‘immediately’, ‘ASAP’ etc. in such a case our model should identify these words and priorities the tickets.
* **Sentiment Grading**: Other scope can be using sentiment analysis to analyze text entered by the user and identify the degree of sentiment expressed and decide dissatisfied users. By doing so we can assign tickets to experience people to handle such users and thus keep the overall satisfaction level intact.
* Other than expanding the scope on the Application side, we can think of improving scope by trying **different word embedding methods**. In our project we have used the GloVe word embedding method during AI model building and we got a fair accuracy score. We can also explore other latest word embedding techniques like FastText, Deep Contextualized Word Representations while building models for Automated Question answering and Sentiment grading.

**Limitations:**

* As observed the given dataset is highly imbalanced for group zero and there is limited data available for the rest of the classes.
* Addition to the imbalance dataset, another issue is the number of training

samples. In our dataset we have 8500 examples including training and test sets.

These numbers are not sufficient for DM models which generally requires data in

Millions.

* We are also not sure about measurement errors. One error can be assigning wrong class labels to many examples. If these misclassifications are observed in minority-class then our model will not be able to predict the correct group during production run.

**References:**

[file:///C:/Users/siddhartha%20borpuzari/Downloads/NLP\_Capstone\_Project\_Solution%20FINAL-1.html](file:///C:\Users\siddhartha%20borpuzari\Downloads\NLP_Capstone_Project_Solution%20FINAL-1.html)

[file:///C:/Users/siddhartha%20borpuzari/Downloads/NLP\_Capstone\_Project\_Solution-Final%20GRU.html](file:///C:\Users\siddhartha%20borpuzari\Downloads\NLP_Capstone_Project_Solution-Final%20GRU.html)