FINAL PROJECT REPORT

AUTOMATIC TICKET ASSIGNMENT USING NLP

Mrinalini Singh, Parmanand Kumar, Sandheep Gopinath

Contents

PROBLEM STATEMENT

One of the key activities of any IT function is to “Keep the lights on” to ensure there is no impact to the Business operations. IT leverages Incident Management process to achieve the above Objective. An incident is something that is unplanned interruption to an IT service or reduction in the quality of an IT service that affects the Users and the Business.

The main goal of Incident Management process is to provide a quick fix / workarounds or solutions that resolves the interruption and restores the service to its full capacity to ensure no business impact. In most of the organizations, incidents are created by various Business and IT Users, End Users/ Vendors if they have access to ticketing systems, and from the integrated monitoring systems and tools. Assigning the incidents to the appropriate person or unit in the support team has critical importance to provide improved user satisfaction while ensuring better allocation of support resources.

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

VALUE PROPOSITION

In the support process, incoming incidents are analyzed and assessed by organization’s support teams to fulfil the request. In many organizations, better allocation and effective usage of the valuable support resources will directly result in substantial cost savings. Currently the incidents are created by various stakeholders (Business Users, IT Users and Monitoring Tools) within IT Service Management Tool and are assigned to Service Desk teams (L1/ L2 teams). This team will review the incidents for right ticket categorization, priorities and then carry out initial diagnosis to see if they can resolve. Around ~54% of the incidents are resolved by L1 / L2 teams. In case L1 / L2 is unable to resolve, they will then escalate / assign the tickets to Functional teams from Applications and Infrastructure (L3 teams). Some portions of incidents are directly assigned to L3 teams by either Monitoring tools or Callers / Requests. L3 teams will carry out detailed diagnosis and resolve the incidents. Around ~56% of incidents are resolved by Functional / L3 teams. In case if vendor support is needed, they will reach out for their support towards incident closure. 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. Around ~25% of Incidents are wrongly assigned to functional teams. Additional effort needed for Functional teams to re-assign to right functional groups. During this process, some of the incidents are in queue and not addressed timely resulting in poor customer service. Guided by powerful AI techniques that can classify incidents to right functional groups can help organizations to reduce the resolving time of the issue and can focus on more productive tasks.

DATA

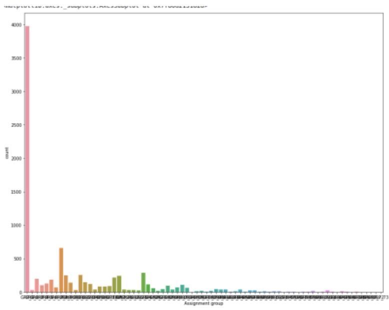
The data set used is directly available in the form of an excel file and hence it can be directly read in the Python code. The data is based on the IT Ticket resolutions that has happened in X Company.
The data contains a few attributes as explained below.

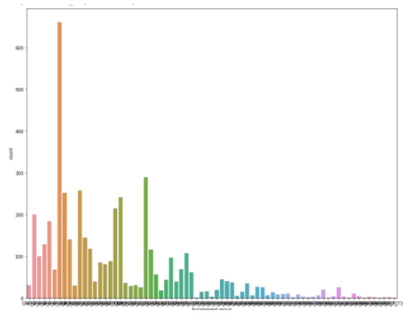
- **Short Description:** It contains a brief of the Ticket explained in a few words. This is similar to the Subject on an Email.
- **Description**: It contains detailed explanation of the issue face. It contains information like the email ID of the person who has raised a request, Details of the issue faced, troubleshooting steps which is done by the user etc. This column contains data in English, German and also contains some invalid characters and Blank values. This is similar to the content of an E-mail.
- **Caller**: This contains information about the person who has raised the request. The data here is encoded or protected.
- **Assignment Group:** This is the dependent variable which gives us information about the group to which the ticket was finally assigned to.

As mentioned above, the data contains Invalid characters, different languages, Blank values etc. These are handled during the pre-processing steps.

- The dataset contains a total of 8500 datapoints.
- Assignment Group is the target group
- The target group contains 74 different Assignment Groups
- The data is not balanced. GRP_0 has the highest number of data points. (3976 data points)
- The frequency of the top 10 groups are as in the below table.

GRP_0	3976
GRP_8	661
GRP_24	289
GRP_12	257
GRP_9	252
GRP_2	241
GRP_19	215
GRP_3	200
GRP_6	184
GRP_13	145





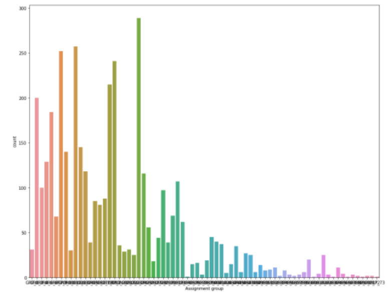


Fig 1.   Fig 1.   Fig 3.

Following are the insides :-

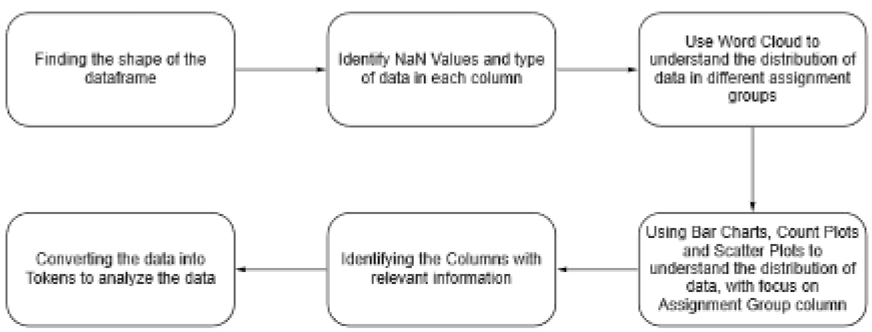
- It is clearly visible that GRP_0 is biasing the data in Fig 1.
- To understand the spread of data, we have plotted the second graph by removing GRP_0 and we could see that GRP_8 is still biasing the data
- On removing GRP_8 we have plotted the third graph and can see that the data is a little more even.

PROCESS OVERVIEW

The code is divided into three Jupyter Notebooks.

- The first part contains Exploratory Data Analysis and Pre-Processing.
    - In this notebook, we have covered the EDA of the dataset
    - Data Cleaning methods
    - Data Pre-processing
        - In Data Pre-Processing, we have done two different approaches and the results are saved into two separate CSV files.
- The Pre-Processed csv files are then further evaluated using different models in Second and Third Notebooks
- The approach followed in Second and Third Notebooks are the same. However, they are done on different datasets to understand the performance of the model on these datasets.

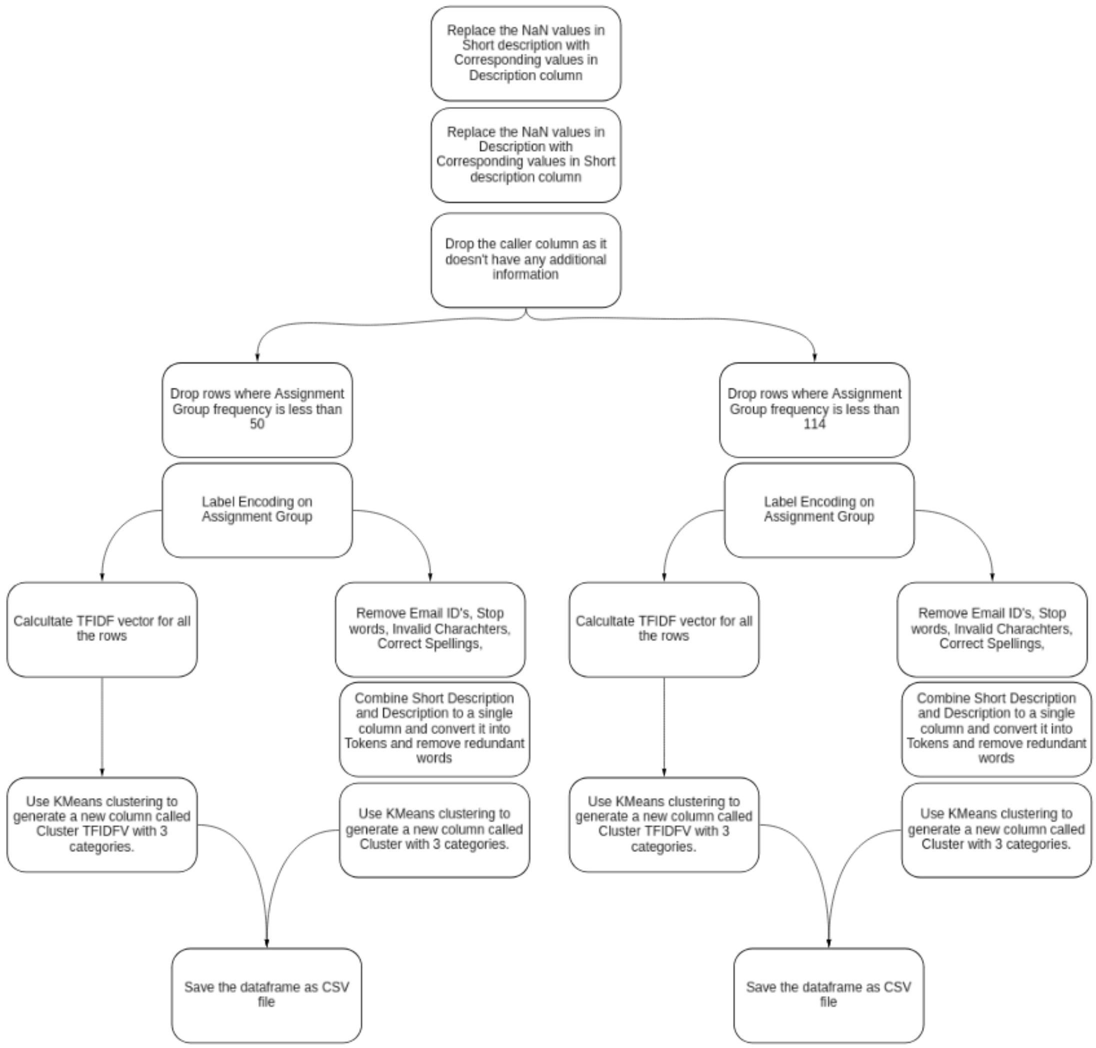
1.Exploratory Data Analysis



The above flowchart explains the Exploratory Data Analysis done on the dataset. Here we have used different methods to

- Identify the shape of the data
- To identify NaN values
- We have used different visualizations to understand the distribution of data
- We have also identified that the Caller Column in not important for the analysis.

2. Data Pre-Processing



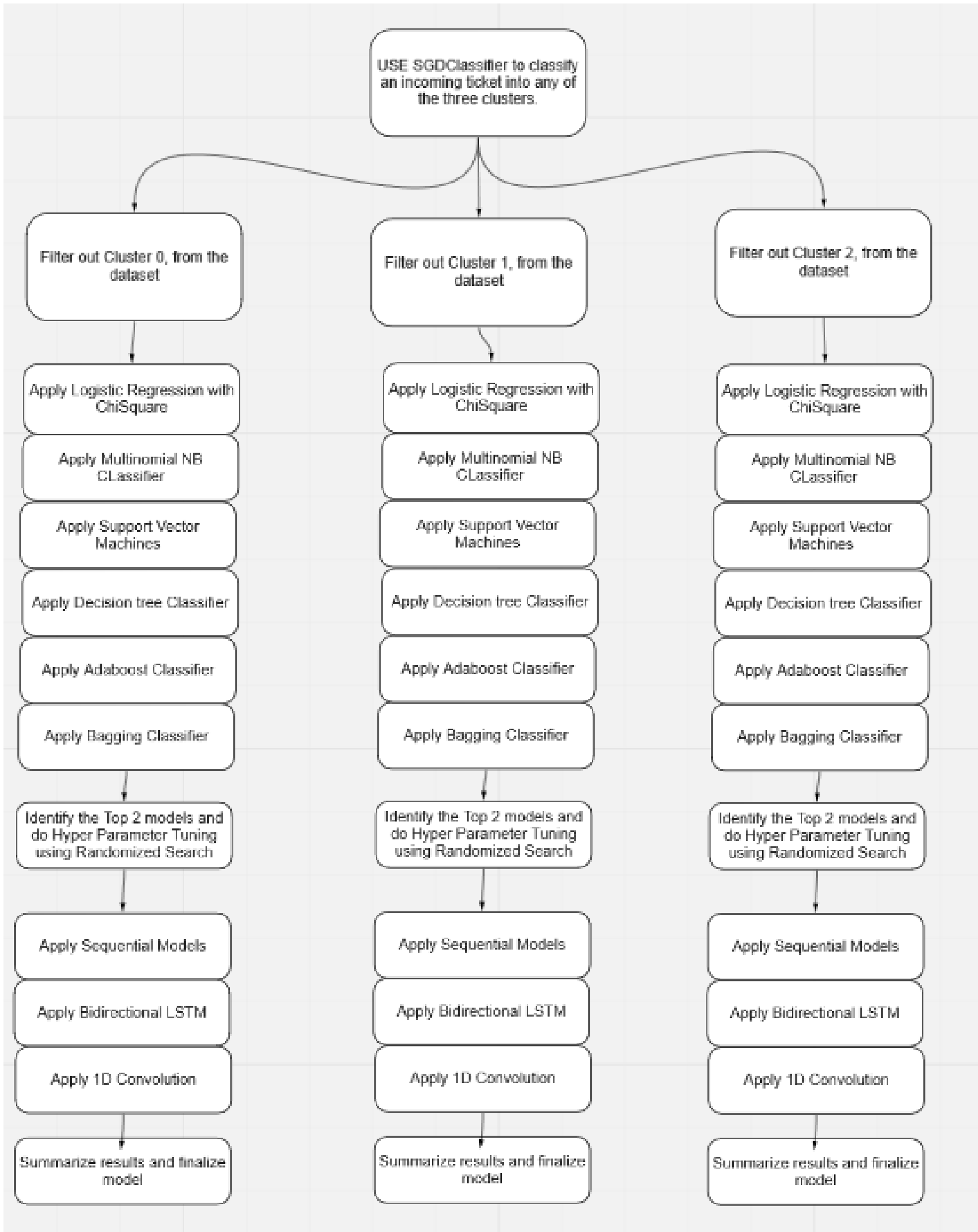
In Data Pre-Processing, we have followed the above flow. We have

- Identified that Short Description has 8 NaN values and Description has 1 NaN values
- Since the content in these columns are always same, we have copied the values of Short Description to Description and Vice versa so that we do not loose any information
- We have also dropped the Caller Column, which we found that we not adding value to analysis.
- We have followed to approached after this , where
  - In the first approach we filtered out and removed Assignment groups with frequency less than 50
  - In Second approach, we removed Assignment groups with frequency less than 114.
  - This approach helped to make the data more clean.

- On both these filtered datasets, we have applied the same data Pre-Processing mechanisms.
    - Label Encoding was done to Encode the assignment groups.
    - We applied TFIDF Vectorizer to convert the Tokens into Vectors
    - We also did this process manually by
        - Tokenizing
        - Removing the Email ID's
        - Removing Invalid characters
        - By correcting Misspelled words
        - Removing stop words.
        - Calculating Term frequency
        - Calculating the TFIDF Vectors using manual formula
- We then applied K-Means clustering to classify this data into three subgroups.
- Clustering helped us to create a Sub Group, and it helps to minimize the number of Unique Assignment groups. By this method, we are able to apply more specific models into each clusters to get better accuracy .

3. Model Building

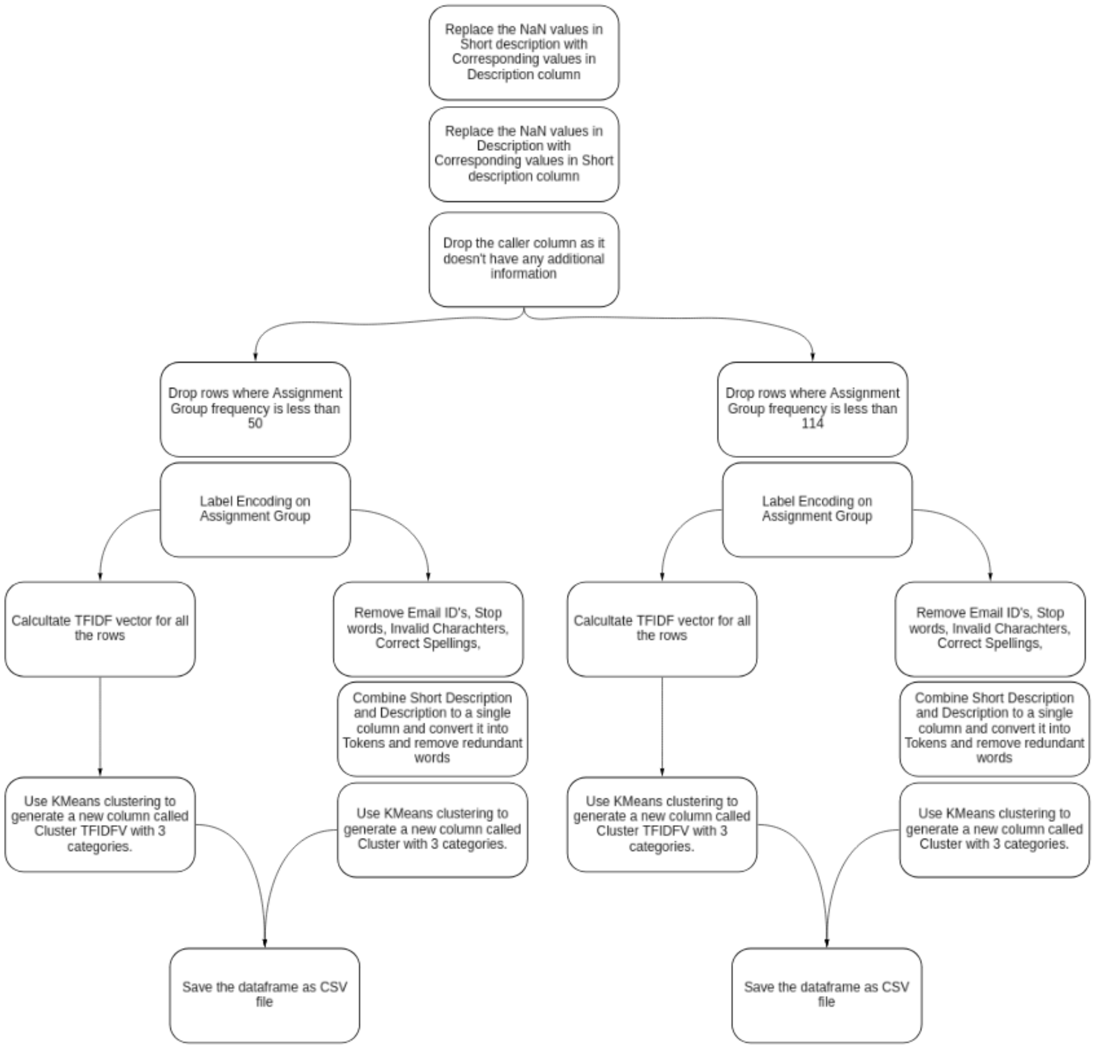
On the above pre-processed datasets, we have applied different models as per the below flowchart.



- We apply SGD Classifier on an incoming ticket to classify it into any of the three groups.
- We classify the groups based on the Cluster and then apply multiple models and analyse the performance.

DATA PRE-PROCESSING : EXPLAINED

The data pre-processing is done is done as per the below flowchart

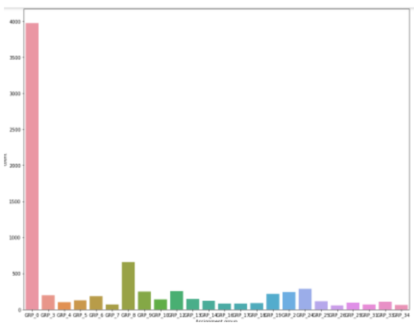


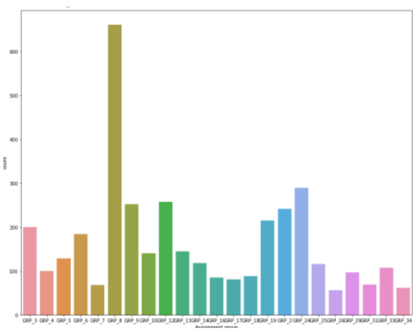
As in the flowchart, we are processing the data in two approaches. In the first approach we are removing the data points which has assignment groups with frequency less than 50.

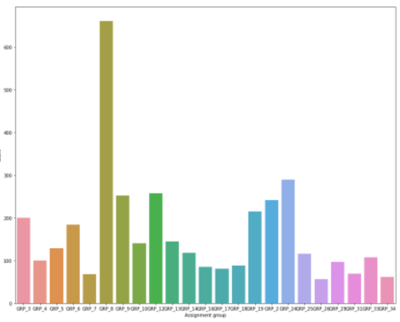
In second approach, we are removing the data points which has assignment groups with frequency less than 114.

On further processing and model building we will be doing the same process on both the files generated out of this process to check the efficiency of both.

DISTRIBUTION OF ASSIGNMENT GROUPS OF REMOVING GROUPS WITH FREQUENCY LESS THAN 50

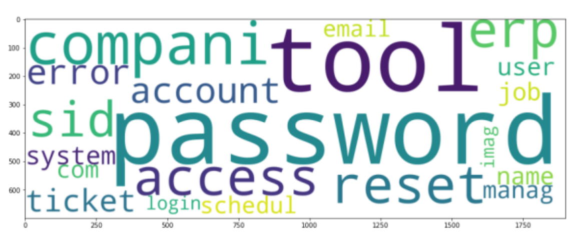




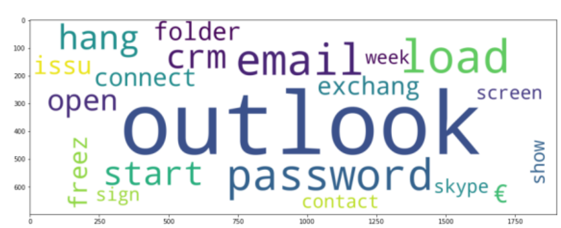


- It is clearly visible that GRP_0 is biasing the data in Fig 1.
- To understand the spread of data, we have plotted the second graph by removing GRP_0 and we could see that GRP_8 is still biasing the data
- On removing GRP_8 we have plotted the third graph and can see that the data is a little more even
- We can see that the data is now more clean.

CLUSTERS GENERATED USING KMEANS with TFIDF Vectorizer



Cluster 1 WordCloud

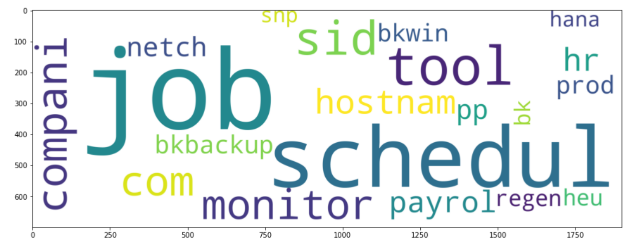


Cluster 2 WordCloud

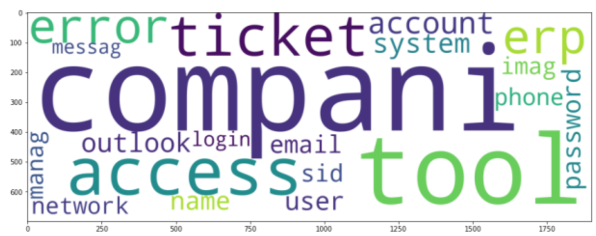
Cluster 3 Wordcloud

CLUSTERS GENERATED USING KMEANS with Manual Vectorizer

Cluster 1 WordCloud



Cluster 2 WordCloud



Cluster 3 Wordcloud

Understanding the clusters

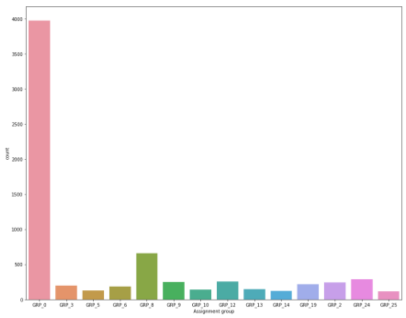
1. Cluster 1, is focused with issues related to Password, Reset, ERP, Login , Account, Windows related issues

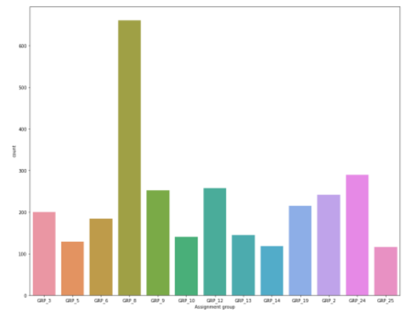
2. Cluster 2 is more focussed with issues related to Job Scheduler, Hostnames, Payrolls, Backups, phones etc

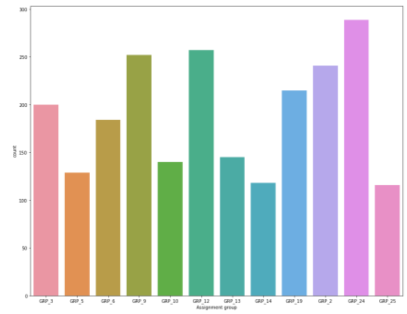
3. Cluster 3 is a more generic group with generic issues. It has issues related to Password, Tools, Access, Phone, Network Login etc

4. The clusters generated using both the process are similar

DISTRIBUTION OF ASSIGNMENT GROUPS OF REMOVING GROUPS WITH FREQUENCY LESS THAN 114

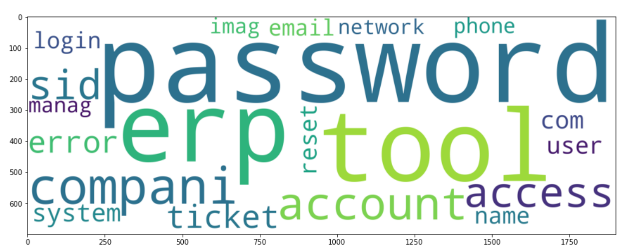




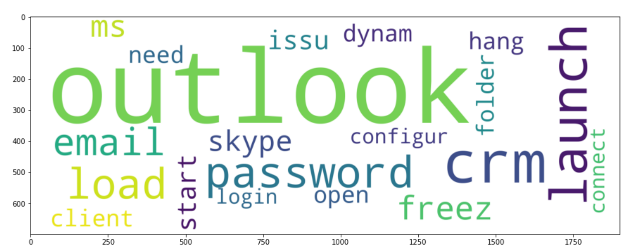


- It is clearly visible that GRP_0 is biasing the data in Fig 1.
- To understand the spread of data, we have plotted the second graph by removing GRP_0 and we could see that GRP_8 is still biasing the data
- On removing GRP_8 we have plotted the third graph and can see that the data is a little more even
- We can see that the data is now even more clean.

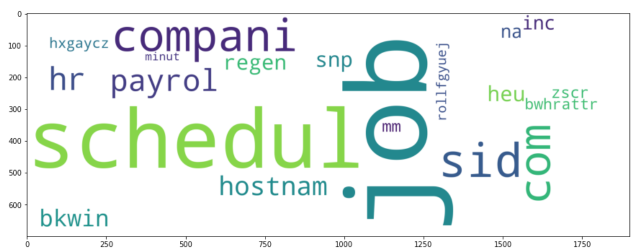
CLUSTERS GENERATED USING KMEANS with TFIDF Vectorizer



Cluster 1 WordCloud

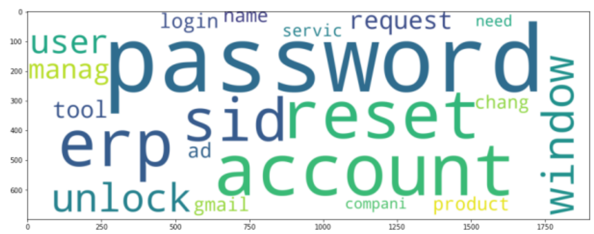


Cluster 2 WordCloud

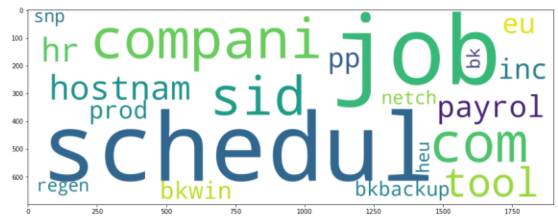


Cluster 3 Wordcloud

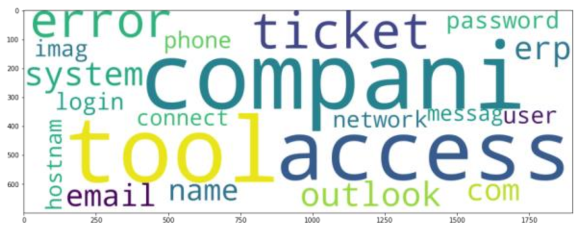
CLUSTERS GENERATED USING KMEANS with Manual Vectorizer



Cluster 1 WordCloud



Cluster 2 WordCloud



Cluster 3 Wordcloud

Understanding the clusters

1. Cluster 1, is focussed with issues related to Password, Reset, ERP, Login , Account, Windows related issues

2. Cluster 2 is more focussed with issues related to Job Scheduler, Hostnames, Payrolls, Backups, phones etc

3. Cluster 3 is a more generic group with generic issues. It has issues related to Password, Tools, Access, Phone, Network Login etc

4. The clusters generated using both the process are similar.

Findings from the Preprocessing

On removing Assignment groups with frequency less than 114, we get a more clean dataset

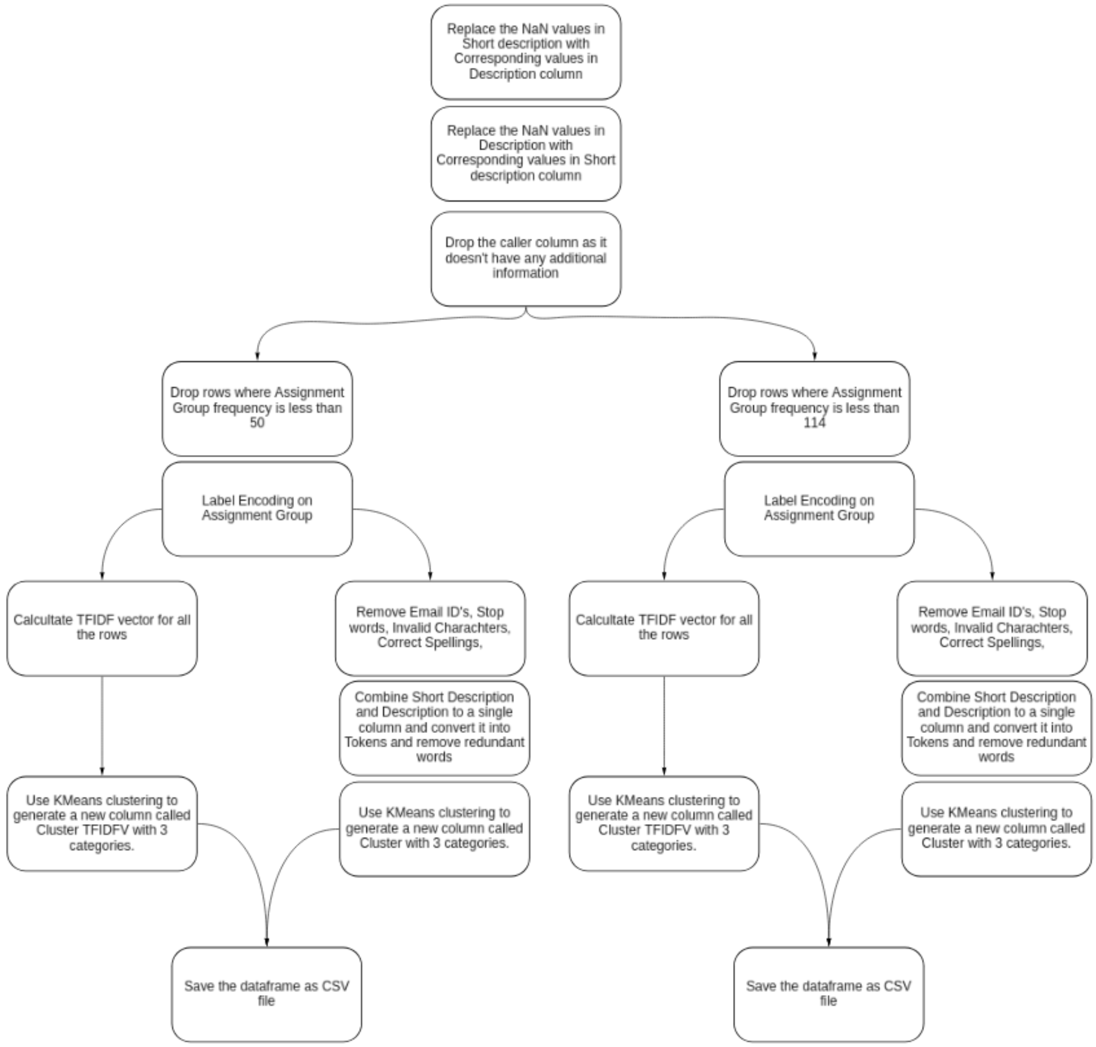
The clusters formed from both the exercises are similar .

The efficiency of the two clusters are to be evaluated during the model building process

While using the inbuilt functions for TFIDF Vectorization and by doing the process manually we get similar clusters

TFIDF Vectorizer is easy to be implemented into pipelines. Since we are getting similar results we will be using that for upcoming stages

MODEL BUILDING



The above approach has been used for the Model building on two datasets .The corresponding results on each cluster are explained in the tables below. The same exercise is done on two different datasets and hence there are two tables for each stage. The first table show the results for dataset 1 and second table shows results for dataset 2

For Classification into cluster

For cluster classification, we have used an SDG Classifier with 98.5% accuracy.

```
sgd = Pipeline([('tfidf', TfidfVectorizer()),
                ('clf', SGDClassifier(loss='log', penalty='elasticnet',alpha=1e-5, random_state=42, max_iter=35, tol=None)),
               ])
sgd.fit(X_train, Y_train)

Y_pred = sgd.predict(X_test)
print(accuracy_score(Y_pred, Y_test))
```

```
0.9850674373795761
```

Classification Models for Cluster 1

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	96.296296	97.142857
1	Multinomial Naive Bayes	96.296296	97.142857
2	Support Vector Machines	96.296296	98.095238
3	Decision Tree	95.555556	98.571429
4	Bagging Classifier	95.555556	98.412698
5	Adaboost Classifier	94.814815	96.031746
6	Gradient Boost Classifier	94.444444	99.841270
7	Decision Tree Hyper Tuned	95.925926	97.936508
8	Support Vector Machines- Hypertuned	96.296296	97.142857
9	Sequential Model	97.777778	98.888886
10	Bidirectional LSTM Model	97.407407	96.666664
11	1D Convolution Model	97.407407	96.666664

Here out of all the models, Hyper tuned SVM has given the highest accuracy and lowest Overfit. On performing the similar analysis on Dataset 2 we get the below results

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	94.444444	91.169451
1	Multinomial Naive Bayes	94.444444	91.169451
2	Support Vector Machines	95.000000	94.510740
3	Decision Tree	94.444444	95.704057
4	Bagging Classifier	93.888889	94.033413
5	Adaboost Classifier	94.444444	93.317422
6	Gradient Boost Classifier	93.333333	99.522673
7	Decision Tree Hyper Tuned	94.444444	93.078759
8	Support Vector Machines- Hypertuned	94.444444	91.169451
9	Sequential Model	91.111112	92.840093
10	Bidirectional LSTM Model	91.111112	92.601430
11	1D Convolution Model	91.111112	92.601430

In this case, a SVM machine gives the best accuracy of 95% on Test dataset with lowest Overfit.

Classification Models for Cluster 2

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	41.911765	46.687697
1	Multinomial Naive Bayes	53.676471	56.782334
2	Support Vector Machines	68.750000	81.072555
3	Decision Tree	47.426471	53.470032
4	Bagging Classifier	44.852941	49.842271
5	Adaboost Classifier	44.117647	50.315457
6	Gradient Boost Classifier	68.382353	85.962145
7	Decision Tree Hyper Tuned	58.088235	92.902208
8	Support Vector Machines- Hypertuned	75.735294	96.056782
9	Sequential Model	61.029410	74.290222
10	Bidirectional LSTM Model	50.735295	42.902207
11	1D Convolution Model	70.220590	77.444798

Here out of all the models, 1D Convolution model has given the highest accuracy and lowest Overfit. On performing the similar analysis on Dataset 2 we get the below results

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	41.970803	46.165884
1	Multinomial Naive Bayes	49.635036	56.651017
2	Support Vector Machines	67.883212	83.411581
3	Decision Tree	48.540146	54.616588
4	Bagging Classifier	46.350365	49.139280
5	Adaboost Classifier	45.620438	49.452269
6	Gradient Boost Classifier	64.233577	86.071987
7	Decision Tree Hyper Tuned	50.364964	86.854460
8	Support Vector Machines- Hypertuned	73.722628	95.774648
9	Sequential Model	56.934309	74.021912
10	Bidirectional LSTM Model	45.985401	44.444445
11	1D Convolution Model	61.313868	75.430357

In this case, a Hyper tuned SVM gives the best accuracy of 67% on Test dataset but with some overfitting on train set

Classification Models for Cluster 3

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	64.080834	62.922604
1	Multinomial Naive Bayes	70.078227	69.460743
2	Support Vector Machines	77.053455	89.354568
3	Decision Tree	67.014342	67.449008
4	Bagging Classifier	65.971317	64.850517
5	Adaboost Classifier	67.275098	66.275496
6	Gradient Boost Classifier	73.859192	91.366303
7	Decision Tree Hyper Tuned	71.056063	91.254540
8	Support Vector Machines- Hypertuned	78.031291	98.463258
9	Sequential Model	68.383312	96.814752
10	Bidirectional LSTM Model	67.535853	80.748814
11	1D Convolution Model	72.359842	98.127967

Here out of all the models, SVM Classifier has given the highest accuracy and lowest Overfit. On performing the similar analysis on Dataset 2 we get the below results

	Model	Accuracy(Test)	Accuracy(Train)
0	Logistic Regression with Chi Square	55.042918	56.186753
1	Multinomial Naive Bayes	60.622318	62.258510
2	Support Vector Machines	70.708155	87.143514
3	Decision Tree	58.798283	61.085557
4	Bagging Classifier	56.652361	57.796688
5	Adaboost Classifier	57.725322	59.084637
6	Gradient Boost Classifier	65.987124	93.905244
7	Decision Tree Hyper Tuned	62.392704	82.244710
8	Support Vector Machines- Hypertuned	73.122318	98.068077
9	Sequential Model	62.982833	94.641215
10	Bidirectional LSTM Model	60.729611	73.712051
11	1D Convolution Model	63.304722	96.573138

In this case, a Hyper tuned SVM gives the best accuracy of 73% on Test dataset but with some overfitting on train set.

COMPARISON TO BENCHMARK

Since we couldn’t find a similar dataset being evaluated online, we are doing our benchmarking against our results as produced from two different datasets.

Based on the above tables, we have the following conclusions

For the clusters,

	Dataset 1	Dataset 2
Cluster classification	98.5%	98.5%
Cluster 0	96%	95%
Cluster 1	70%	67%
Cluster 2	77%	73%

From the above comparison, we can see that the model performs better when operated on Dataset 1.

In Dataset1, we have dropped more number of assignment groups, whose frequency was less than 114.

Hence it clearly shows that for certain assignment groups we did not have enough data and the need to collect more data points in such a way that the dataset is biased and has sufficient information

VISUALIZATIONS

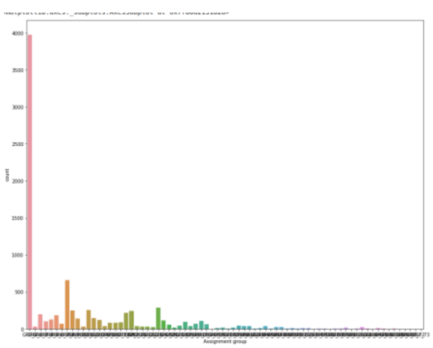


Fig 1

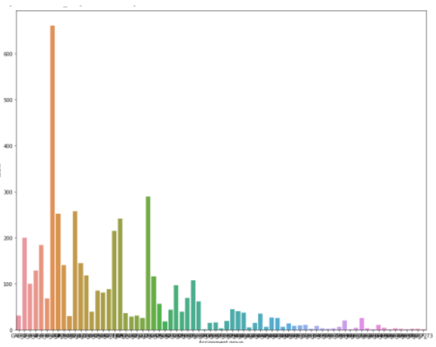


Fig 2

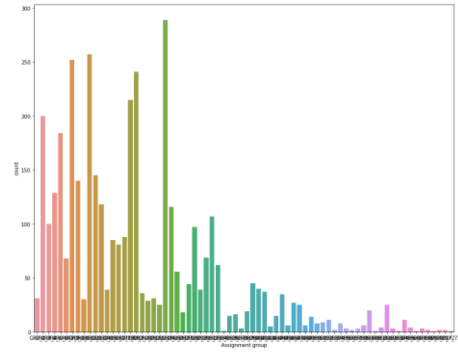
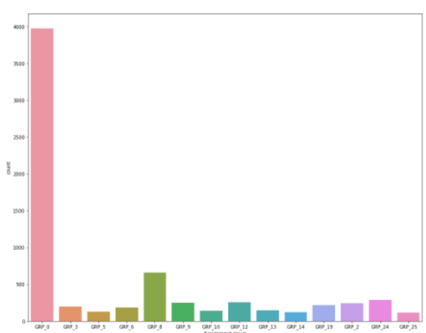
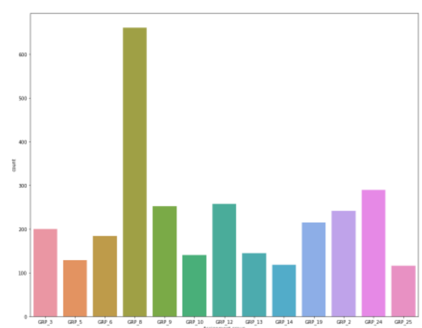
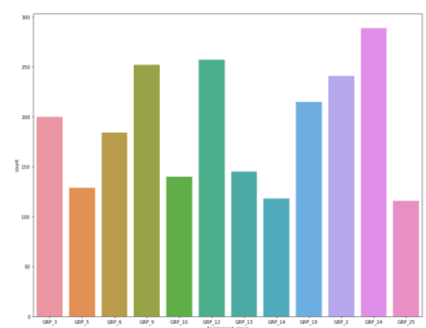


Fig 3

The above visualization represents how the data was distributed in the raw data. It show that the data is clearly biased and we have a lot of groups which have extremely low data points to create a model. It is based on these visualizations that we decided to drop the groups which doesn't have enough information. And we can see from the above comparison that Dataset 1 was giving better results. That is when we have dropped all the datapoints which was occurring less than 114 times. And the data looks as in the below graphs.





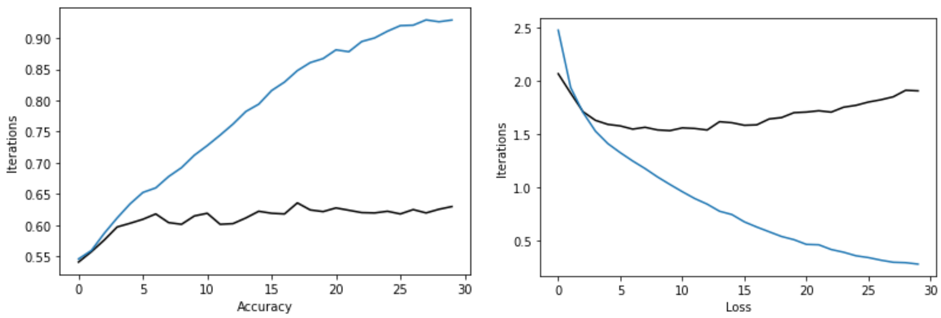


We can see that the database is now more clean and doesn't have groups which are occurring less than 114 times.

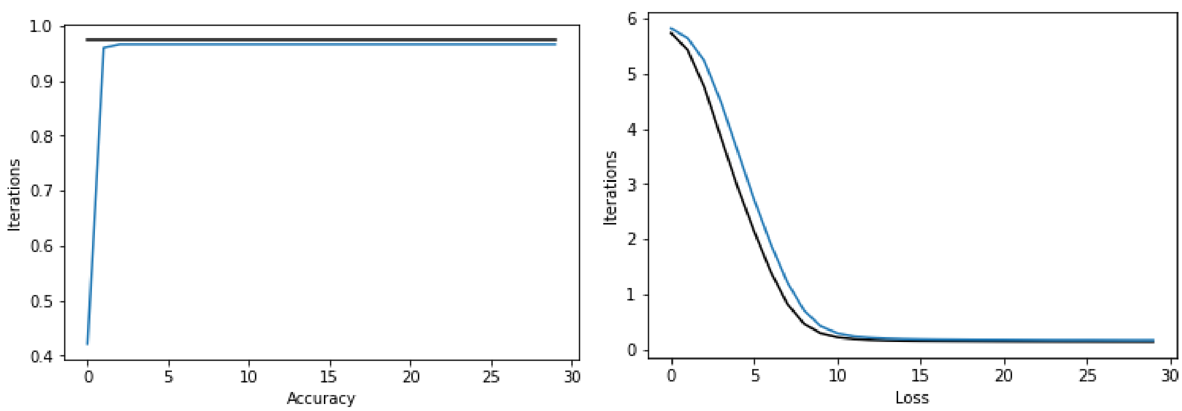
Below are some training visualizations for the deep learning models for Dataset 1

For Cluster 1

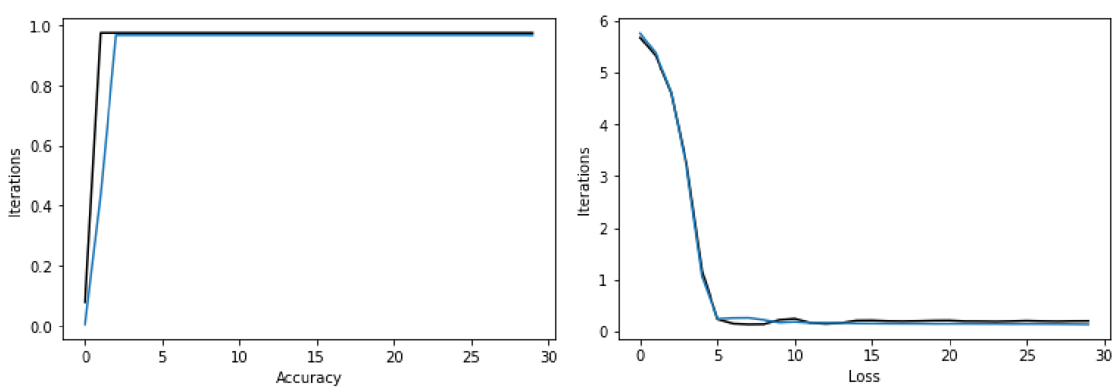
Training graphs over iterations for Sequential Model



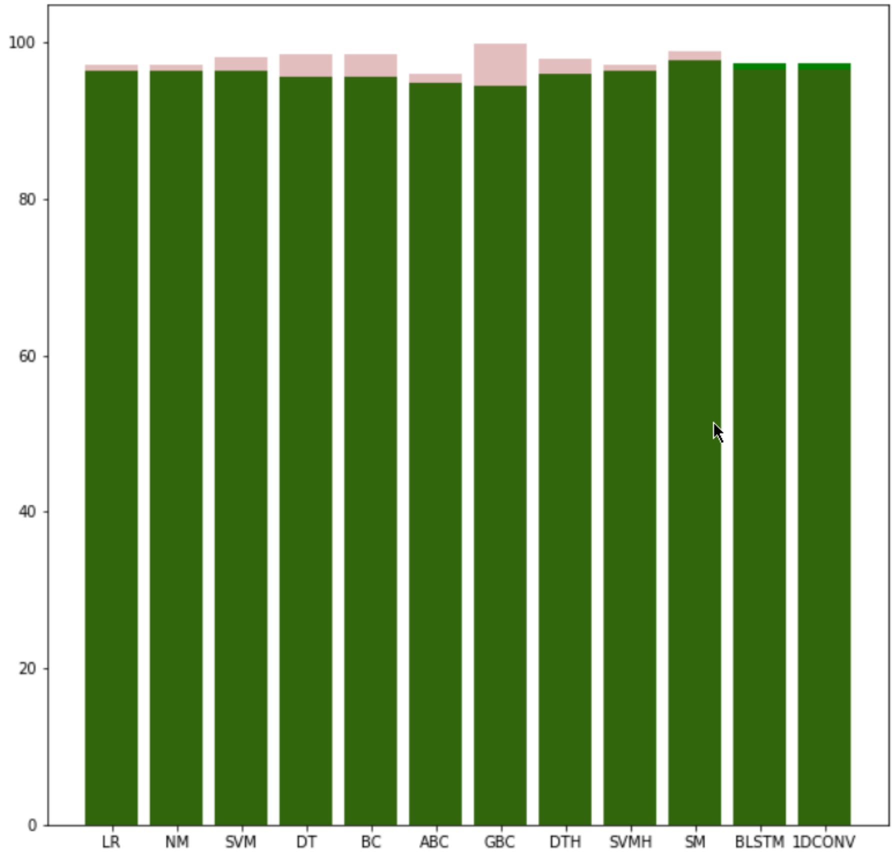
Training graphs over iterations for Bi-Directional LSTM



Training graphs over iterations for 1D Convolution



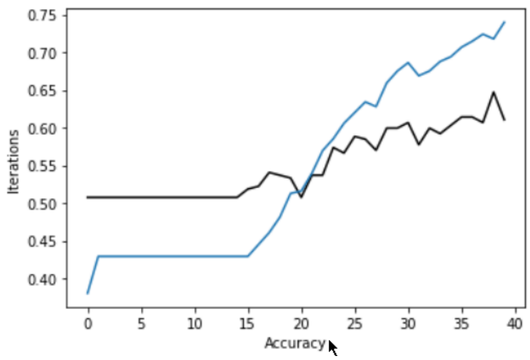
Representation of All model performances

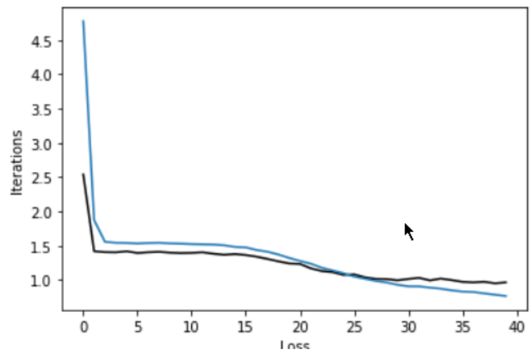


Here we can see that the Hypertuned SVM has the highest accuracy and lowest overfitting of all the models .

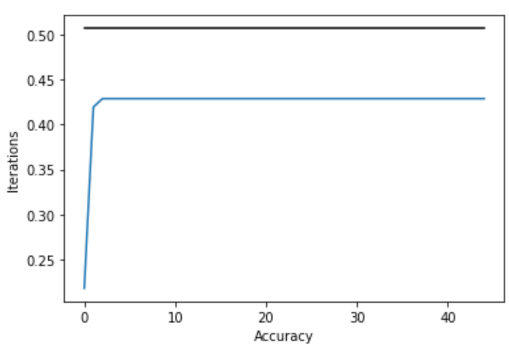
For Cluster 2

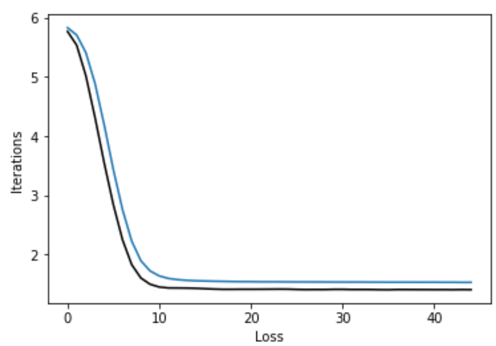
Training graphs over iterations for Sequential Model



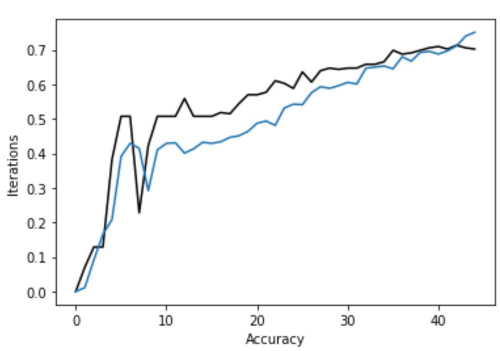


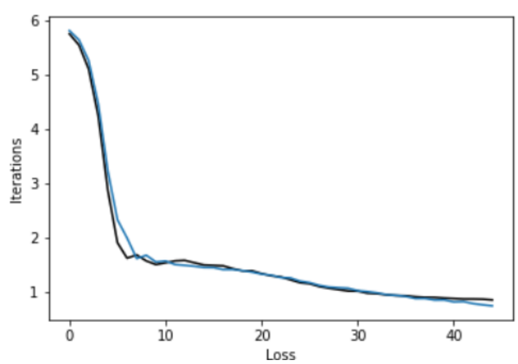
Training graphs for Bidirectional LSTM



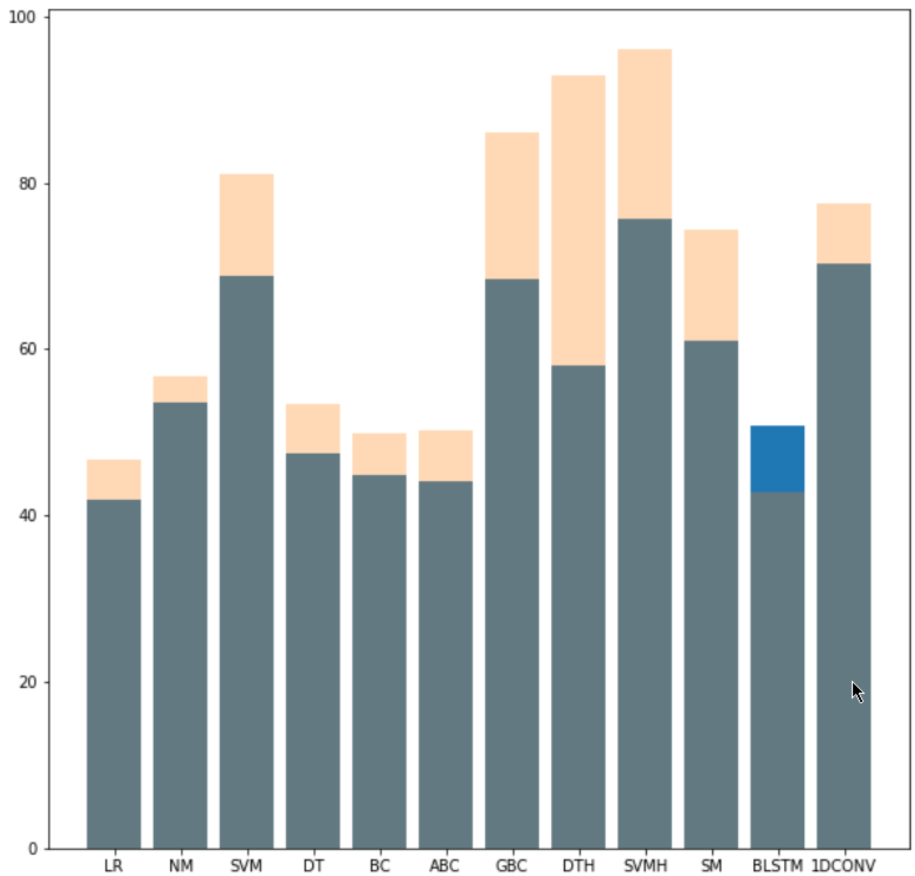


Training graphs over iterations for 1D Convolution





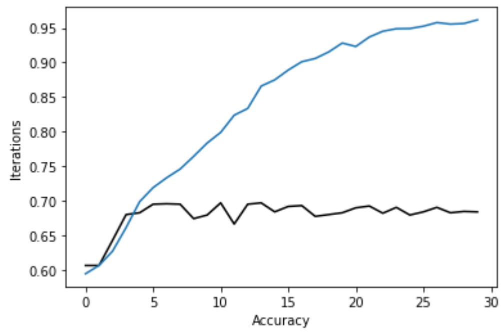
Representation of All model performances

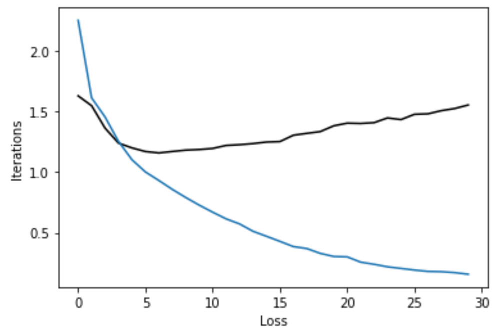


Here we can see that the 1D convolution has the highest accuracy and lowest overfitting of all the models .

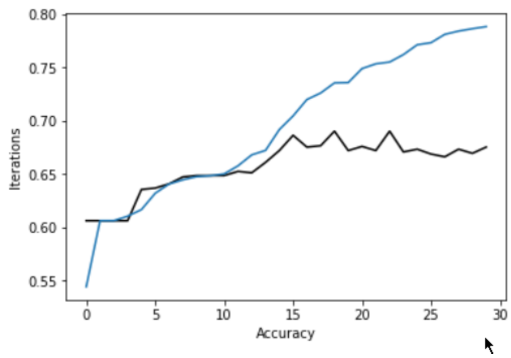
For Cluster 3

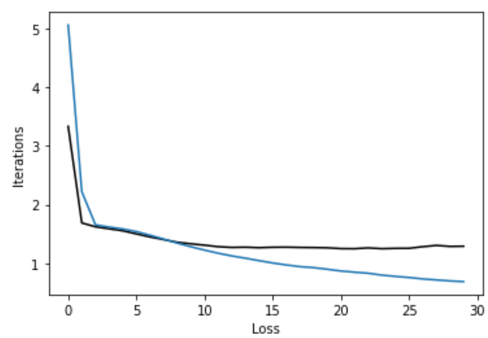
Training graphs over iterations for Sequential Model



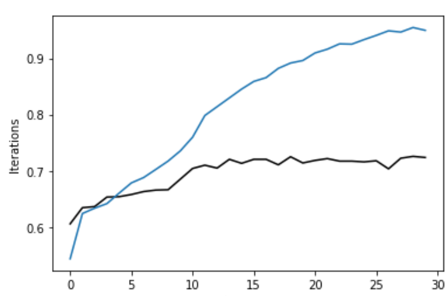


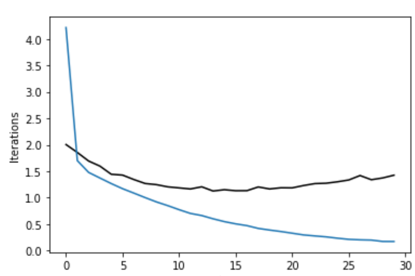
Training graphs for Bidirectional LSTM



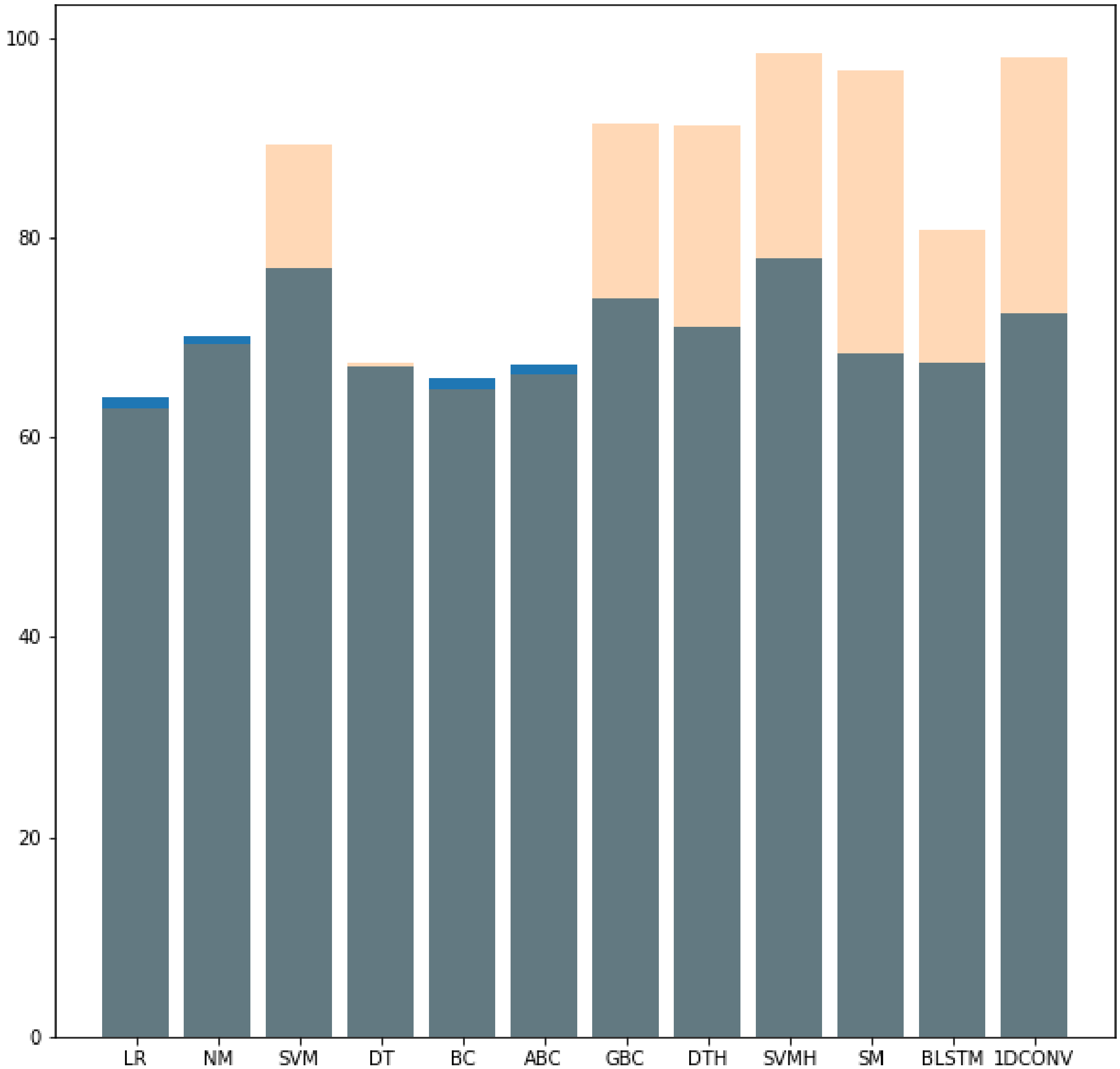


Training graphs over iterations for 1D Convolution





Representation of All model performances



Here we can see that the SVM Classifier has the highest accuracy and lowest overfitting of all the models .

IMPLICATIONS

In the IT Ticket support process, incoming incidents are analyzed and assessed by organization’s support teams to fulfill the request. In many organizations, better allocation and effective usage of the valuable support resources will directly result in substantial cost savings. Currently the incidents are created by various stakeholders (Business Users, IT Users and Monitoring Tools) within IT Service Management Tool and are assigned to Service Desk teams (L1/ L2 teams). This team will review the incidents for right ticket categorization, priorities and then carry out initial diagnosis to see if they can resolve. Around ~54% of the incidents are resolved by L1 / L2 teams. Incase L1 / L2 is unable to resolve, they will then escalate / assign the tickets to Functional teams from Applications and Infrastructure (L3 teams). Some portions of incidents are directly assigned to L3 teams by either Monitoring tools or Callers / Requestors. L3 teams will carry out detailed diagnosis and resolve the incidents. Around ~56% of incidents are resolved by Functional / L3 teams. Incase if vendor support is needed, they will reach out for their support towards incident closure. 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. Around ~25% of Incidents are wrongly assigned to functional teams. Additional effort needed for Functional teams to re-assign to right functional groups. During this process, some of the incidents are in queue and not addressed timely resulting in poor customer service. Guided by powerful AI techniques that can classify incidents to right functional groups can help organizations to reduce the resolving time of the issue and can focus on more productive tasks.

The advantage of our approach is that we have a multilevel prediction system.

In the first level, we arxe able to classify the incoming ticket into any of the 3 clusters with an accuracy of 98.5%. This helps in assigning the ticket to the right experts within IT and solves the problem if assigning the ticket to wrong group.

Now to assign the ticket to a right team, we are using 3 different models. Based on our results we will be able to classify a ticket to the end team with a confidence level of 70%-90%. However, this can further be improved by using a bigger dataset to train the data.

However, this is not a generic solution and will perform differently in different organizations as theory structure might be different. In the current use case, where the data is taken from we will be able to classify the data into the end user with the above-mentioned accuracies.

LIMITATIONS

We had also conducted some surveys in an organisation where they have a similar pain point. The finding was that the initial prediction of our model, which groups that data into any of the 3 predefined cluster will help the local IT in saving time in segregating the tickets. However, the end team to which the team is assigned to may vary from case to case.

EG: If someone raised an issue that ‘Outlook is not working”, based on our model , it will predict that the ticket is to be assigned to Cluster 2, which has a defined number of end teams, who might be able to solve this.
Here again, the issue can be due to multiple issues.

- It might be because the users password has expired. (It can be sorted by local IT)
- It can be because the users VPN is not connected (can be sorted by local IT)
- It can be because the network connection is poor (To be handled by network team)
- It can be because the Outlook server is down(To be handled by server team)

Here, we can see that the same ticket can get assigned to different teams based on the use case. Hence it is not possible to automate the entire system by using our model. Our model can make the process faster by assigning it to a cluster, who might be able to solve the issue.
In order to completely automate the system, we will have it include automated tested pipelines, to check for the above issues automatically and then intelligently assign the ticket.

CLOSING REFLECTIONS

All though there are above mentioned restrictions which will stop up from fully automating the system , but the model can help in reducing the time taken to assign the ticket. As mentioned earlier, when a ticket is raised, it comes to the local IT in most organisations. The local IT analyse the ticket and then assigns it to the respective teams.

We have faced issues in this process. If the local IT is busy with other tasks, often these assignments get delayed and hence cause other business impacts, which are severe some times. Our process helps to eliminate this delay and hence to automate the process.
But still , based on our dataset, we had to eliminate some tickets as we didn’t have enough data for all groups. In order to make this more efficient, we will have to collect more data from the end user and retrain the model so that it can become more efficient in production.