AUTOMATIC TICKET ASSIGNMENT

MINI PROJECT REPORT

Real problem and Business value

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.

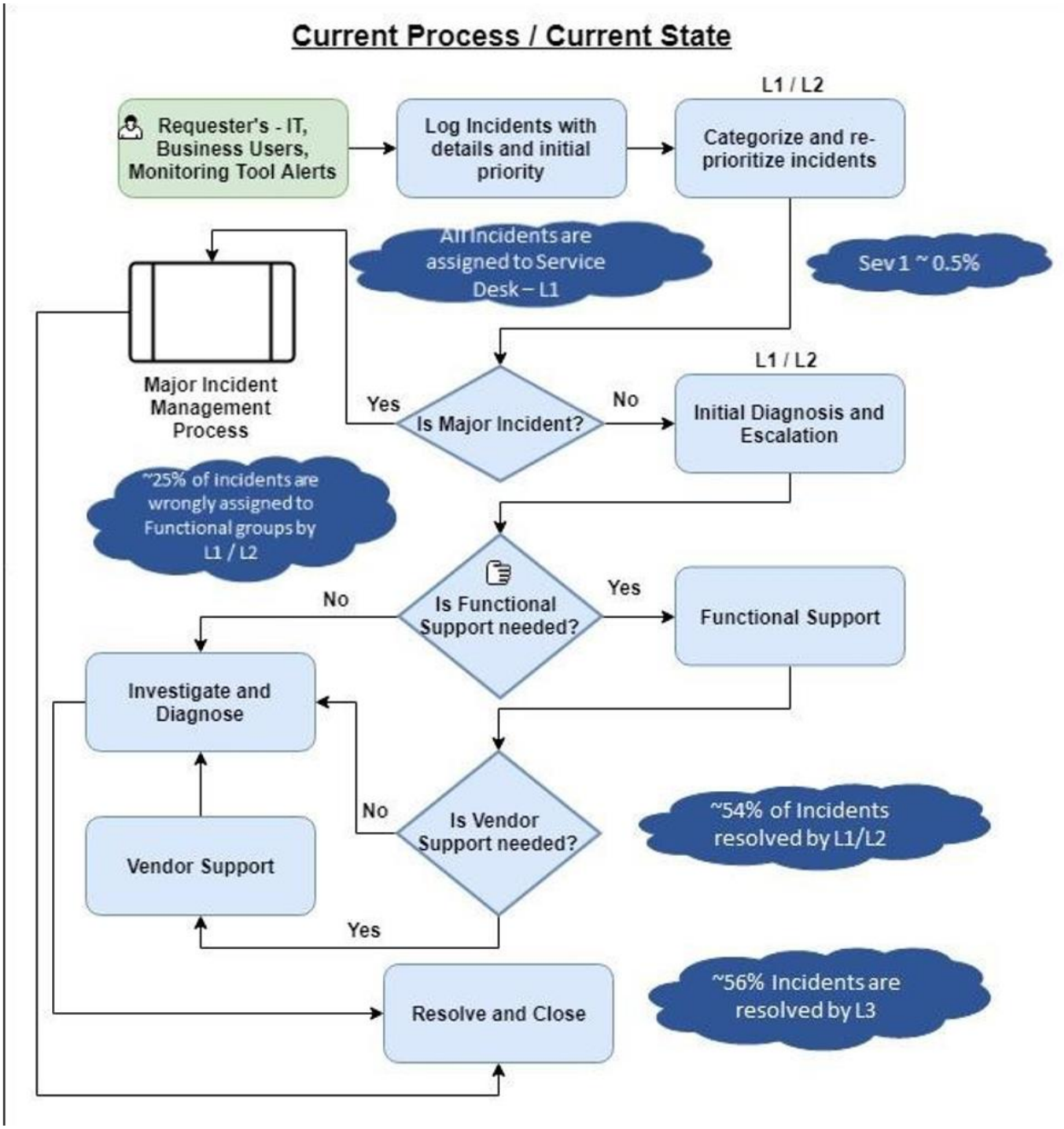
Manual assignment of incidents is time consuming and requires human efforts. There may be mistakes due to human errors and resource consumption is carried out ineffectively because of the misaddressing. On the other hand, manual assignment increases the response and resolution times which result in user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

Business Domain Value

In the support process, incoming incidents are analyzed and assessed by the 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 the 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. If 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 need to be reviewed for SOPs before ticket assignment). 15 min is being spent on SOP review for each incident. Minimum of ~1 FTE effort needed only for incident assignment to L3 teams.



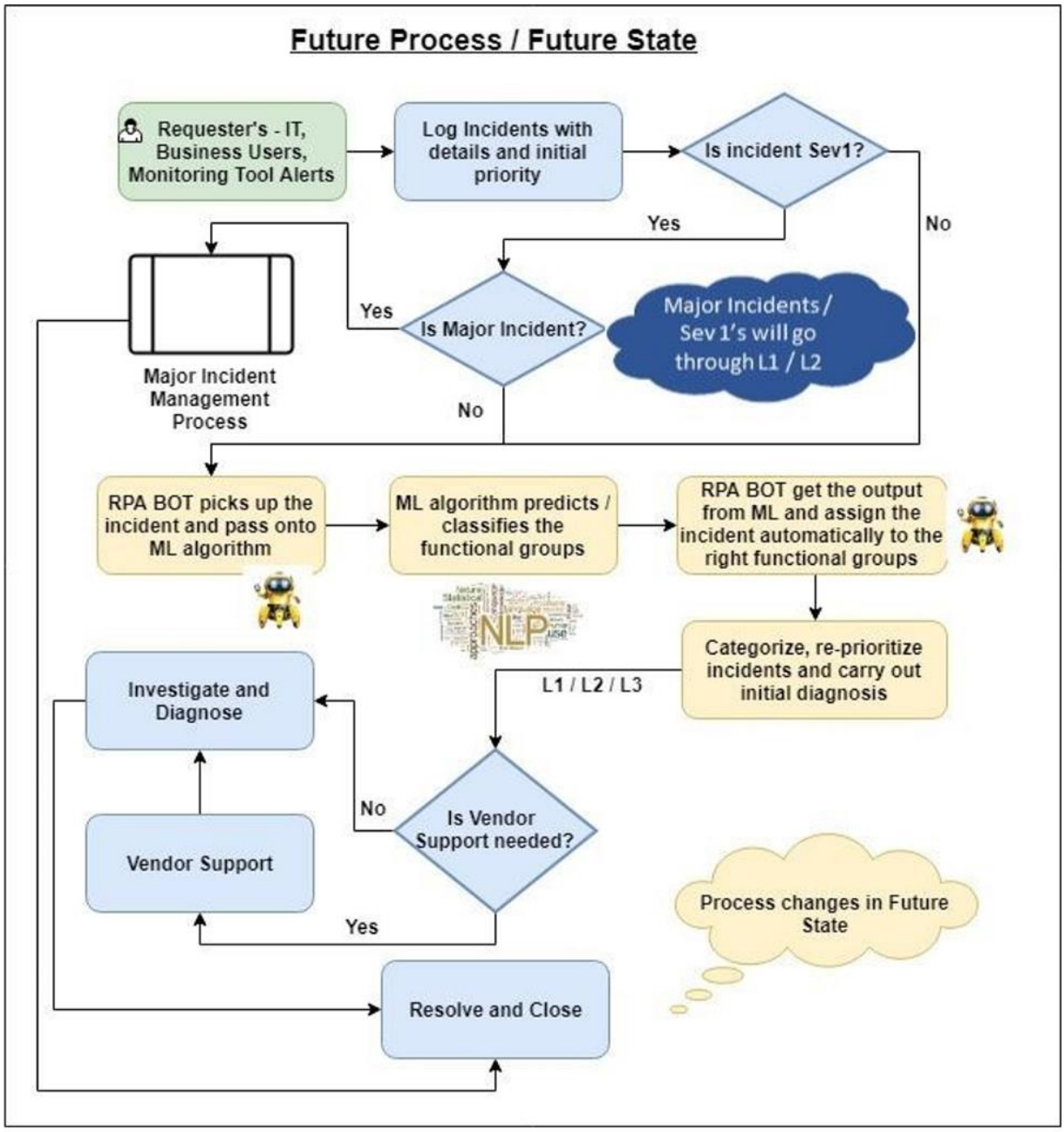
Summary of problem statement

The objective of the problem classifies the target group, given the short and long description. Which helps the business in solving the tickets issue in time without delay which leads to business impact.

Future State / Proposed Solution / Solution Architecture

Functional Architecture:

- Shows how the new Incident Management process will look like once the IT Service Management Tool is integrated with RPA and Machine Learning model / solution.
- The incidents will continue to be created by various stakeholders (Business Users, IT Users and Monitoring Tools) within IT Service Management Tool.
- Once the Incident is created in the IT Service Management tool, there would be a scheduled RPA BOT that will pick up those Incidents one by one, get all the required features and pass onto the Machine Learning model.
- The output from the Machine Learning model will be the predicted “Assignment Group” that is passed onto the RPA BOT.
- The RPA BOT will then go the IT Service Management Tool and assign the Incident to that Assignment Group coming from Machine Learning model.
- Then the process of Incident resolution continues as defined in the current state process.
- Any exceptions to this automated process will be handled by the L1 / L2 teams manually.


Future Process / Future State
Requester's - IT, Business Users, Monitoring Tool Alerts
Log Incidents with details and initial priority
Is incident Sev1?
Yes
No
Yes
Is Major Incident?
Major Incidents / Sev 1's will go through L1 / L2
No
Major Incident Management Process
RPA BOT picks up the incident and pass onto ML algorithm
ML algorithm predicts / classifies the functional groups
RPA BOT get the output from ML and assign the incident automatically to the right functional groups
NLP
Categorize, re-prioritize incidents and carry out initial diagnosis
L1 / L2 / L3
Investigate and Diagnose
No
Is Vendor Support needed?
Vendor Support
Yes
Process changes in Future State
Resolve and Close

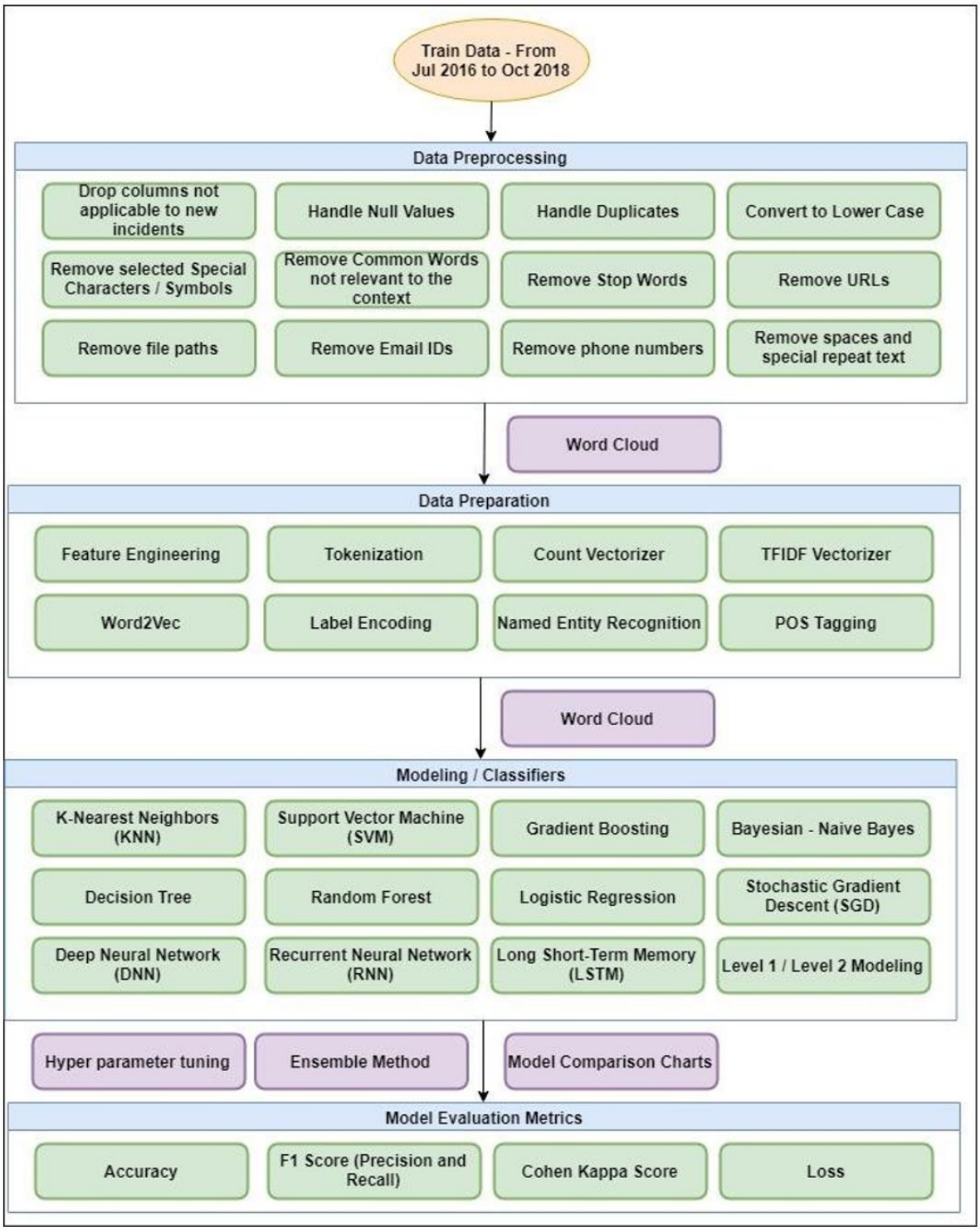
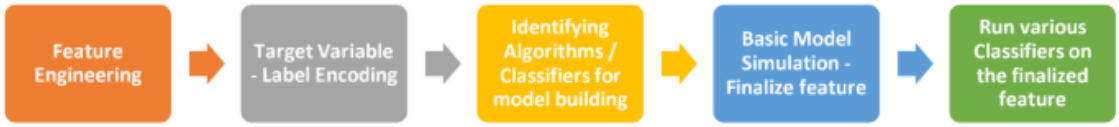

Train Data - From Jul 2016 to Oct 2018
Data Preprocessing
Drop columns not applicable to new incidents
Handle Null Values
Handle Duplicates
Convert to Lower Case
Remove selected Special Characters / Symbols
Remove Common Words not relevant to the context
Remove Stop Words
Remove URLs
Remove file paths
Remove Email IDs
Remove phone numbers
Remove spaces and special repeat text
Word Cloud
Data Preparation
Feature Engineering
Tokenization
Count Vectorizer
TFIDF Vectorizer
Word2Vec
Label Encoding
Named Entity Recognition
POS Tagging
Word Cloud
Modeling / Classifiers
K-Nearest Neighbors (KNN)
Support Vector Machine (SVM)
Gradient Boosting
Bayesian - Naive Bayes
Decision Tree
Random Forest
Logistic Regression
Stochastic Gradient Descent (SGD)
Deep Neural Network (DNN)
Recurrent Neural Network (RNN)
Long Short-Term Memory (LSTM)
Level 1 / Level 2 Modeling
Hyper parameter tuning
Ensemble Method
Model Comparison Charts
Model Evaluation Metrics
Accuracy
F1 Score (Precision and Recall)
Cohen Kappa Score
Loss

Figure-10: Data Preparation pipeline

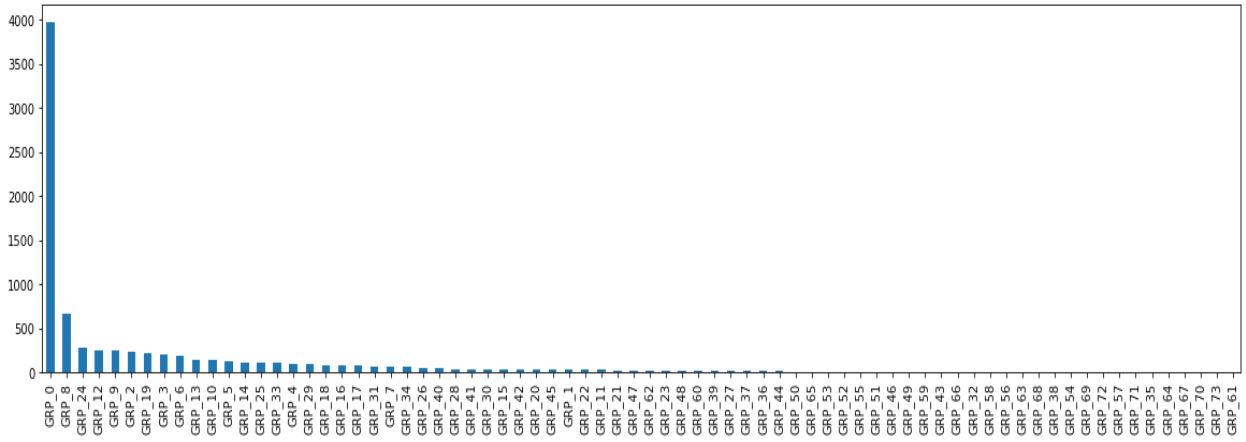


Summary of EDA & Pre-processing

For assigning tickets automatically the data set contains **8500** rows and **4** columns.
Columns: Description, Short Description, Caller & Group,

Basically, it's a classification problem to correctly determine the group for assigning tickets automatically. There are groups from 0 to 74 (target), below is the target class distribution.

Out of the 8498 data 3975 pertains to the group 0 and 661 pertains to group 8 which is almost 50% of the dataset. Rest of the dataset is below 1% and can be avoided. Many assignment groups have only one test data which can be deleted. Some of the cases it is seen that the description is not in English language and as the data is very few so that can also be deleted. Below is the plot for Assignment group with their count.



Here our challenge is to build a model which has a higher accuracy in spite of the data mainly consisting of only a few groups from the 74 groups.

Pre-processing

2.1 Lower case

The first pre-processing step which we will do is transform our tweets into lower case. This avoids having multiple copies of the same words. For example, while calculating the word count, ‘Analytics’ and ‘analytics’ will be taken as different words.

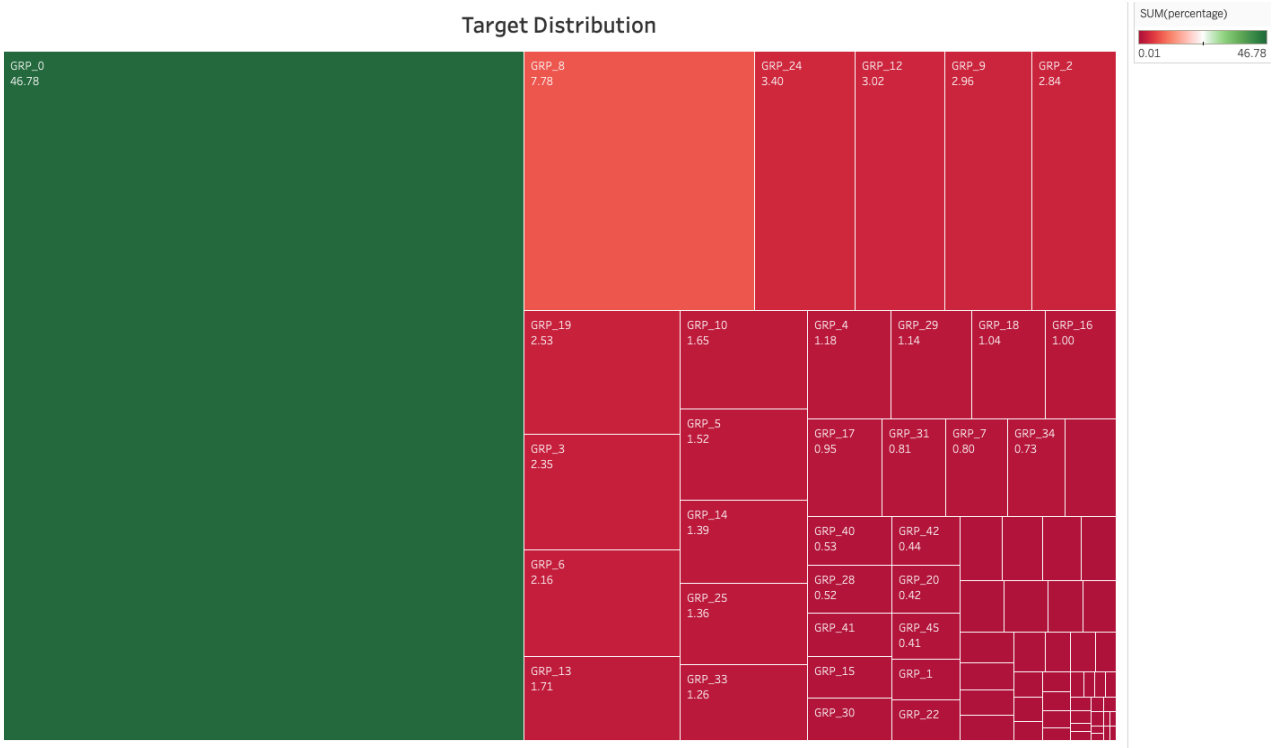
2.2 Removing Punctuation

The next step is to remove punctuation, as it doesn’t add any extra information while treating text data. Therefore, removing all instances of it will help us reduce the size of the training data.

As you can see in the above output, all the punctuation, including ‘#’ and ‘@’, has been removed from the training data.

2.3 Removal of Stop Words

Stop words (or commonly occurring words) should be removed from the text data. For this purpose, we can either create a list of stop words ourselves or we can use predefined libraries.



2.4 Common word removal

Previously, we just removed commonly occurring words in a general sense. We can also remove commonly occurring words from our text data First, let's check the 10 most frequently occurring words in our text data then take a call to remove or retain.

2.6 Spelling correction

We've all seen tweets with a plethora of spelling mistakes. Our timelines are often filled with hasty sent descriptions that are barely legible at times.

In that regard, spelling correction is a useful pre-processing step because this also will help us in reducing multiple copies of words. For example, "Analytics" and "analytics" will be treated as different words even if they are used in the same sense.

To achieve this we will use the *textblob* library.

2.8 Stemming

Stemming refers to the removal of suffices, like "ing", "ly", "s", etc. by a simple rule-based approach. For this purpose, we will use *PorterStemmer* from the NLTK library.

2.9 Lemmatization

Lemmatization is a more effective option than stemming because it converts the word into its root word, rather than just stripping the suffices. It makes use of the vocabulary and does a morphological analysis to obtain the root word. Therefore, we usually prefer using lemmatization over stemming.

Features reduction:

As the caller is not having any significance in model building, we have removed it straight away, then we applied Fuzzy Wuzzy and compared the similarities between short description and description and a new column has been created called **"Derived Description",** if the similarity is more than 90% derived description will contain only short description**.**

Target class reduction:

As we seen in the above image the targets are not equally balanced, in face many target classes are not even having in 1% data.So we merged the assignment group where there is less than 1% data to group no:67 and the total assignment groups reduced to 50 from 74.

	Assignment_group	counts	percentage	cumulative_percentage
0	GRP_0	3968	46.731834	46.731834
1	GRP_8	661	7.784713	54.516547
2	GRP_24	289	3.403604	57.920151
3	GRP_12	257	3.026734	60.946885
4	GRP_9	252	2.967848	63.914733
5	GRP_2	241	2.838299	66.753033
6	GRP_19	215	2.532093	69.285125
7	GRP_3	200	2.355435	71.640561
8	GRP_6	184	2.167000	73.807561
9	GRP_13	145	1.707690	75.515251
10	GRP_10	140	1.648805	77.164056
11	GRP_5	129	1.519256	78.683312
12	GRP_14	118	1.389707	80.073018
13	GRP_25	116	1.366152	81.439171
14	GRP_33	107	1.260158	82.699329
15	GRP_4	100	1.177718	83.877046
16	GRP_29	97	1.142386	85.019432
17	GRP_18	88	1.036391	86.055824
18	GRP_16	85	1.001060	87.056884
19	GRP_17	81	0.953951	88.010835
20	GRP_31	69	0.812625	88.823460
21	GRP_7	68	0.800848	89.624308
22	GRP_34	61	0.718408	90.342716
23	GRP_26	56	0.659522	91.002238
24	GRP_40	45	0.529973	91.532211

So, we converted the less popular target classes to single calls “less_pop” this lead us to reduce the target classes to **50.**

Word cloud

We created the word clouds before text cleaning and after cleaning.

Before cleaning:

After text cleaning:

Model Building:

Before creating the model there are certain steps like the target variable y, we used label encoder and converted to integers and for the X we have tokenized, performed count vectorizer for converting it to vector.

Models Evaluation:

Random Forest Ensemble Model Technique:

	precision	recall	f1-score	support
accuracy			0.66	1700
macro avg	0.29	0.20	0.21	1700
weighted avg	0.60	0.66	0.60	1700

Total Accuracy:0.65

Decision Tree Ensemble Model Technique:

	precision	recall	f1-score	support
accuracy			0.56	1700
macro avg	0.17	0.17	0.16	1700
weighted avg	0.53	0.56	0.54	1700

Total Accuracy:0.56

Ada Boost Ensemble Model Technique:

	precision	recall	f1-score	support
accuracy			0.49	1700
macro avg	0.01	0.02	0.01	1700
weighted avg	0.25	0.49	0.32	1700

Total Accuracy:0.49

Logistic Regression Model Technique:

	precision	recall	f1-score	support
GRP_0	0.75	0.91	0.82	950
GRP_1	0.50	0.33	0.40	9
GRP_3	0.88	0.51	0.65	41
GRP_4	0.33	0.50	0.40	2
GRP_5	0.54	0.58	0.56	64
GRP_6	0.58	0.54	0.56	39
GRP_7	0.73	0.23	0.35	35
GRP_8	0.80	0.40	0.53	10
GRP_9	0.64	0.36	0.46	25
GRP_10	1.00	0.95	0.98	22
GRP_11	0.48	0.43	0.45	23
GRP_12	0.39	0.37	0.38	54
GRP_13	0.56	0.54	0.55	61
GRP_14	0.00	0.00	0.00	9
GRP_15	0.00	0.00	0.00	6
GRP_16	0.00	0.00	0.00	10
GRP_17	0.43	0.75	0.55	4
GRP_18	0.90	0.87	0.88	70
GRP_19	0.76	0.37	0.50	35
GRP_2	0.29	0.20	0.24	10
GRP_20	0.00	0.00	0.00	5
GRP_21	0.67	0.13	0.22	15
GRP_22	0.59	0.65	0.62	20
GRP_23	0.50	0.39	0.44	56
GRP_24	0.29	0.15	0.20	13
GRP_25	0.22	0.12	0.16	16
GRP_26	0.35	0.29	0.32	24
GRP_27	0.50	0.30	0.38	23
GRP_28	0.33	1.00	0.50	1
GRP_29	0.00	0.00	0.00	6
GRP_30	0.50	0.50	0.50	6
GRP_31	0.58	0.27	0.37	26
GRP_33	0.67	0.33	0.44	12
GRP_34	0.86	0.60	0.71	10
less_pop	0.40	0.25	0.31	8
GRP_36	1.00	0.50	0.67	2
GRP_37	0.00	0.00	0.00	6
GRP_39	0.33	0.11	0.17	9
GRP_40	0.00	0.00	0.00	9
GRP_41	0.73	0.42	0.53	38
GRP_42	0.00	0.00	0.00	3
GRP_44	0.00	0.00	0.00	2
GRP_45	0.66	0.46	0.54	54
GRP_47	0.00	0.00	0.00	4
GRP_48	0.25	0.17	0.20	6
GRP_50	0.00	0.00	0.00	3
GRP_53	0.64	0.27	0.38	26
GRP_60	0.54	0.90	0.67	158
GRP_62	0.48	0.18	0.26	62
GRP_65	0.42	0.24	0.30	21
accuracy			0.67	2123
macro avg	0.44	0.34	0.36	2123
weighted avg	0.64	0.67	0.63	2123

Total Accuracy:0.67

Model Accuracy details

S.No	Model	Train accuracy %	Test accuracy%
1.	Logistic Regression	67	65
2.	Decision Tree	92	57
3.	Ada Boost	46	48
4.	Random Forest	92	66
5.	LSTM	83	60

Upsampling

We did the upsampling to improve the accuracy. After upsampling below are the updated accuracies.

S.No	Model	Train accuracy %	Test accuracy%	F1 Score
6.	Logistic Regression	78	70	71
7.	Decision Tree	94	92	92
8.	Ada Boost	33	50	50
9.	Random Forest	94	92	93

Hyper-Parameter Tuning

- **Optimizers-** Adam, rmsprop, sgd
- **Activation Functions-** Softmax, relu, sigmoid
- **Learning rate-** 0.01 and 0.001
- **Vocab size-** 2000 to 20000
- **Epochs-** 5 to 20
- **Loss function-** sparse_categorical_crossentropy, binary_crossentropy, categorical_crossentropy
- **Metrics-** accuracy

Final Parameters Selection

- **Optimizers-** Adam
- **Activation Functions-** Softmax, relu
- **Learning rate-** 0.01 and 0.001
- **Vocab size-** 2000 to 20000
- **Epochs-** 5 to 20

- **Loss function-** sparse_categorical_crossentropy
- **Metrics-** accuracy

Final Model Selection

Models	accuracy%	val_accuracy%	loss	val_loss	epochs
1. LSTM	82	62	0.69	1.56	10
2. LSTM(glove embedding)	96	97	0.14	0.13	20
3. GRU	93	60	0.24	1.72	20

The final model we selected is “LSTM with glove embedding” with the following:

epochs - 20

Train Accuracy – 96%

Val_accuracy - 97%

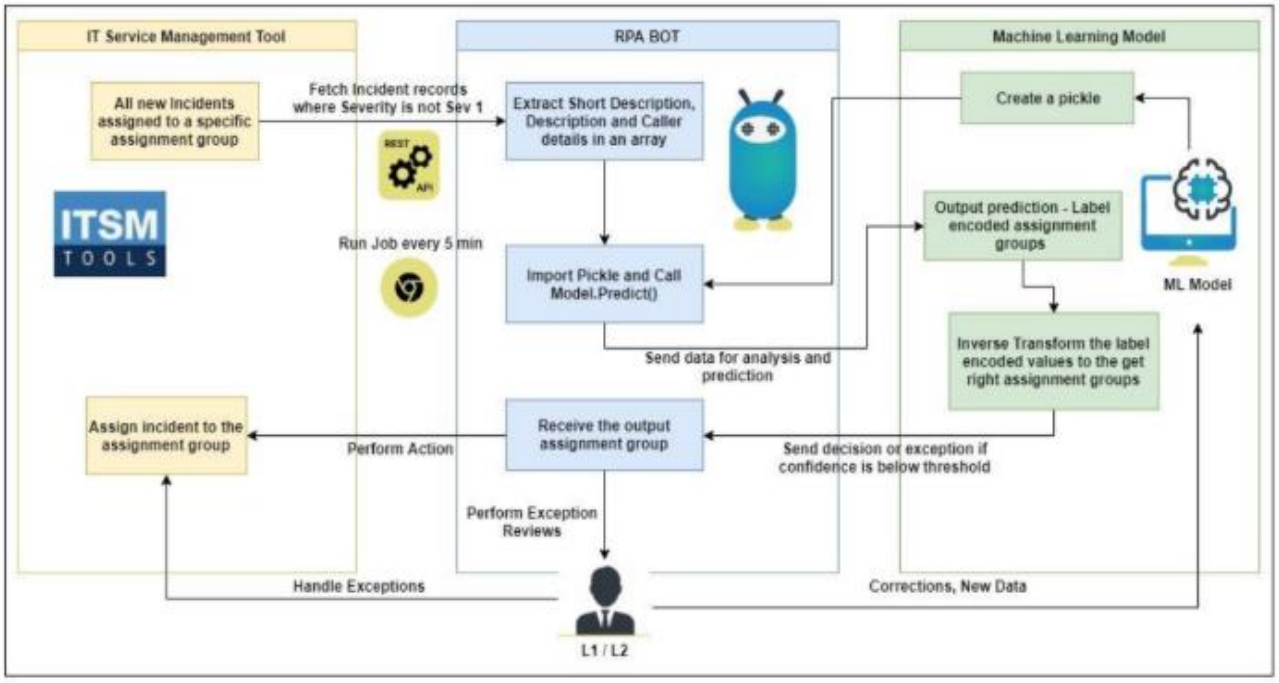
Optimizer - Adam

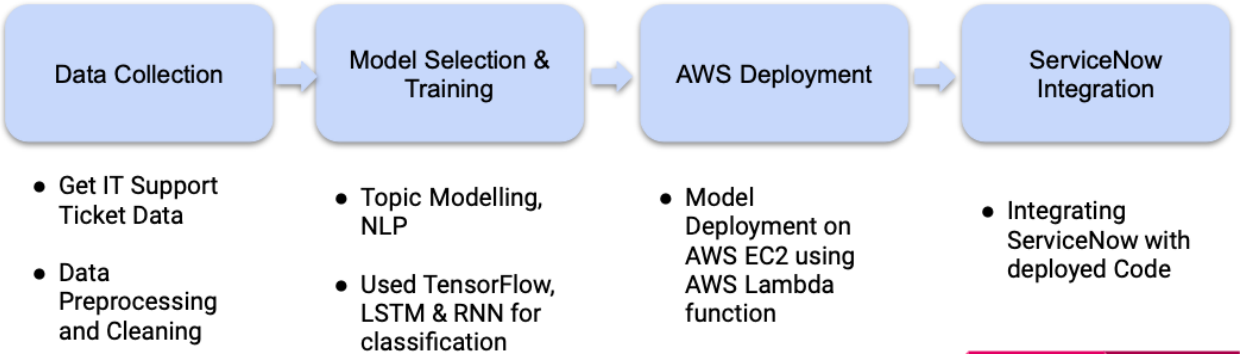
Activation function – softmax

Loss function - sparse_categorical_crossentropy

Metrics – accuracy

Figure-56: Productionizing the Machine Learning Model





Future Scope and Limitations

We have proposed the ticket assignment engine that uses an ensemble of machine learning techniques. 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.

However, there is still some scope for improvement of the system. The system can also be enhanced to handle concept drift better. However, in datasets with high concept drift this method may not give good results over the long run. We can build chatbots.

Questions and Answers related to Project

Q1. What was the size of the data?
Answer: **8500** rows and **4** columns.

Q2. What was the data type?
Answer: Int and Str

Q3. What was the team size and distribution?
Answer: 7

Q4. What techniques were you using for data pre-processing for various data science use cases and visualization?
Answer:

- Converting the text to Lower Case
- Removing Punctuation, special characters.
- Removal of stop words
- Common and rare words removal
- Spelling correction
- Stemming: Stemming refers to the removal of suffices, like “ing”, “ly”, “s”, etc. by a simple rule-based approach. For this purpose, we will use *PorterStemmer* from the NLTK library.
- Lemmatization: Lemmatization is a more effective option than stemming because it converts the word into its root word, rather than just stripping the suffices. It makes use of the vocabulary and does a morphological analysis to obtain the root word.
- Created Word cloud before and after cleaning of the text.
- Features reduction: As the caller is not having any significance in model building we have removed it straight away, Then we applied Fuzzy Wuzzy and compared the similarities between short description and description and a new column has been created called “Derived Description”, if the similarity is more than 90% derived description will contain only short description.
- Target class reduction: As we seen in the above image the targets are not equally balanced, in face many target classes are not even having in 1% data. So we merged the assignment group where there is less than 1% data to group no:67 and the total assignment groups reduced to 50 from 74.

Q5. What were your roles and responsibilities in the project?
Answer: My responsibilities consisted of gathering the dataset, Cleaning, and getting familiar with the data, model training. Performing various EDA and feature engineering techniques and testing by applying various machine learning models. Also performing hyper-parameter Tuning for getting optimized results.

Q6. In which area you have contributed the most?
Answer: I contributed the most to data preparation, preprocessing and model training areas. Also, we did a lot of brainstorming for finding and selecting the best algorithms for our use cases. After that, we identified and finalized the best practices for implementation.

Q7. In which technology you are most comfortable?
Answer: I have worked in almost all the fields viz. Machine Learning, Deep Learning, and Natural Language Processing, and I have nearly equivalent knowledge in these fields. But preferred one is loved working in ML.

Q8. In how many projects you have already worked?
Answer: I have worked in various projects e.g., object detection, object classification, object identification, NLP projects, machine learning regression, and classification problems.

Q9. What kind of challenges have you faced during the project?
Answer: The biggest challenge that we face is in terms of obtaining a good dataset, since it was an imbalanced dataset, cleaning it to be fit for feeding it to a model. Then comes the task of finding the correct algorithm to be used for that business case. Then that model is optimized. If we are exposing the model as an API, then we need to work on the SLA for the API as well, so that it responds in optimum time.

Q10.How did you optimize your solution?
Answer: Model optimization depends on a lot of factors.

- Train with better data (increase the quality),or do data pre-processing steps more efficiently.
- Increase the quantity of data used for training.
- Hyper- parameter tuning performs regularly.

Q11.How much time did your model take to get trained?
Answer: 1 – 2 hrs

Q12. At what frequency are you retraining and updating your model?
Answer: At every 10-15 days

Q15. In which mode have you deployed your model?
Answer: I have deployed the model both in cloud environments as well in the on-premise ones based on the client and project requirements