**Automatic Group Categorization and Real Time Ticket Resolution Prediction**

DISSERTATION

Submitted in partial fulfillment of the requirements of the

MTech Data Science and Engineering Degree program

By

Ashish Sarkar

2019hc04098

Under the supervision of

Mr. Sandip Das, Engagement Portfolio Lead

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**CERTIFICATE**

This is to certify that the Dissertation entitled Automatic Group Categorization and Real Time Ticket Resolution Time Prediction and submitted by Mr. Ashish Sarkar ID. No. 2019hc04098 in partial fulfillment of the requirements of DSECLZG628T Dissertation, embodies the work done by him under my supervision.

A picture containing icon

Description automatically generated

Signature of the Supervisor

Name: Sandip Das

Place: Kolkata Designation: Engagement Portfolio Lead

Date: 27-Feb-2022

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**Acknowledgement**

I would like to thank my supervisor Mr. Sandip Das for his constant guidance. I would also like to thank my examiner Prof. Ramesh Ramani for his support. A special thanks to Birla Institute of Technology & Science, Pilani for opening door to completion of my post-Graduation. Finally, I like to express my deepest gratitude to my family and friends for their unfailing support and continuous encouragement.

A picture containing text, tableware, plate, dishware

Description automatically generated

Ashish Sarkar

Place: Kolkata

Date: 27-Feb-2022

Contents

[Acknowledgement 3](#_Toc96901833)

[Abstract 9](#_Toc96901834)

[1.1 Problem: 9](#_Toc96901835)

[1.2 Business Objective: 10](#_Toc96901836)

[1.3 Automatically predictive support tickets resolution time 10](#_Toc96901837)

[1.4 Benefits: 11](#_Toc96901838)

[1.5 Methodology Used in Classification: 11](#_Toc96901839)

[1.6 Potential challenges & risks in doing the project 13](#_Toc96901840)

[1.7 Methodology Used in Prediction: 13](#_Toc96901841)

[Introduction 15](#_Toc96901842)

[1.8. Feature Vector representation of ticket data: 34](#_Toc96901843)

[1.9 Building Classification models: - 34](#_Toc96901844)

[1.10. Implementation and experimental results: 34](#_Toc96901845)

[1.10.1 Data set analysis and pre-processing 35](#_Toc96901846)

[1.10.2 Fixing Imbalanced Data Set 35](#_Toc96901847)

[1.11. Feature Vector representation of ticket data 36](#_Toc96901848)

[1.12. Building and Evaluation of the models 37](#_Toc96901849)

[1.13. Future Enhancement 39](#_Toc96901850)

[1.14. Conclusion related to classification models 40](#_Toc96901851)

[PART-B: 40](#_Toc96901852)

[1.15. Data Preprocessing: 40](#_Toc96901853)

[1.5.1 Null value check 40](#_Toc96901854)

[1.15.2 Removing outlier 41](#_Toc96901855)

[1.16 Encoding Categorical Values 45](#_Toc96901856)

[1.16.1 Ordinal encoding 45](#_Toc96901857)

[1.17. Improving Machine Learning Models by Adding Features: - 46](#_Toc96901858)

[1.17.1 Start date/time of the ticket: 46](#_Toc96901859)

[1.17.2 Performance of the resolved by: 47](#_Toc96901860)

[23.1 Linear Regression: 56](#_Toc96901861)

[23.2 Random Forest: - 57](#_Toc96901862)

[24. Real time analysis of the performance of the model: 59](#_Toc96901863)

[Conclusion: 61](#_Toc96901864)

[Future Enhancement: 62](#_Toc96901865)

[Bibliography / References: 62](#_Toc96901866)

[Appendices: 62](#_Toc96901867)

[Figure 1:Model Pipeline 8](#_Toc96901868)

[Figure 2 Figure 2:Classification Model Pipeline 12](#_Toc96901869)

[Figure 3:Predictive Model Pipeline 12](#_Toc96901870)

[Figure 4:Assigment group 14](#_Toc96901871)

[Figure 5: Ticket Distribution Group 14](#_Toc96901872)

[Figure 6: Severity Bar Graph 15](#_Toc96901873)

[Figure 7: Category Bar Graph vs Count 15](#_Toc96901874)

[Figure 8: Category vs Actual Time 16](#_Toc96901875)

[Figure 9: Priorities per Category 16](#_Toc96901876)

[Figure 10: Reassignment Count 16](#_Toc96901877)

[Figure 11: Reopen Count 17](#_Toc96901878)

[Figure 12: Reopen Count VS Actual Time 17](#_Toc96901879)

[Figure 13: Reopen Box Plot Count 17](#_Toc96901880)

[Figure 14: SLA vs Actual Time 18](#_Toc96901881)

[Figure 15: Ticket Frequency 18](#_Toc96901882)

[Figure 16: Impact Details 19](#_Toc96901883)

[Figure 17: Impact Details vs Actual Time 19](#_Toc96901884)

[Figure 18: Urgency 20](#_Toc96901885)

[Figure 19: Urgency vs Actual Time 20](#_Toc96901886)

[Figure 20: Priority vs Time 20](#_Toc96901887)

[Figure 21: Priory vs Actual Time 21](#_Toc96901888)

[Figure 22:Distribution of Resolved Time 21](#_Toc96901889)

[Figure 23: Distribution of Actual Time 21](#_Toc96901890)

[Figure 24: Box Plot of Reassignment Count 22](#_Toc96901891)

[Figure 25: Month vs Target vs SLA 22](#_Toc96901892)

[Figure 26: Month Count vs Target 23](#_Toc96901893)

[Figure 27: Day of week vs Target vs SLA 23](#_Toc96901894)

[Figure 28: day of week Violin Plot 24](#_Toc96901895)

[Figure 29: Word Cloud of Unique words 25](#_Toc96901896)

[Figure 30: Word Cloud of Oracle Support Group 25](#_Toc96901897)

[Figure 31: Imbalance Data Set 28](#_Toc96901898)

[Figure 32: Balanced Data Set 29](#_Toc96901899)

[Figure 33: Model Scores 30](#_Toc96901900)

[Figure 34: Model Scores 31](#_Toc96901901)

[Figure 35: Actual Time Distribution and QQ Plot 33](#_Toc96901902)

[Figure 36: Actual Time Distribution 33](#_Toc96901903)

[Figure 37: Actual Time CDF Plot 34](#_Toc96901904)

[Figure 38:Actual Time Box Plot after Outlier Removal 34](#_Toc96901905)

[Figure 39: Actual Time Normally Distributed and QQ Plot 35](#_Toc96901906)

[Figure 40: Comparative study of Actual Time before and after - outlier removal 35](#_Toc96901907)

[Figure 41: Mean Average Time 38](#_Toc96901908)

[Figure 42: Histogram Representation of each feature 39](#_Toc96901909)

[Figure 43: SNS Plot of feature variable vs Target 40](#_Toc96901910)

[Figure 44: ExtraTreeclassifier Feature Selection 41](#_Toc96901911)

[Figure 45: Random Forest Classification 41](#_Toc96901912)

[Figure 46: Linear Regression Plot 48](#_Toc96901913)

[Figure 47: Actual vs Prediction 48](#_Toc96901914)

[Figure 48: RF Regression Line 49](#_Toc96901915)

[Figure 49: Actual vs Prediction plot for RF 49](#_Toc96901916)

[Figure 50: Deployed Model Architect ure 51](#_Toc96901917)

[Figure 51: UI Screen 1 51](#_Toc96901918)

[Figure 52: UI Screen 2 51](#_Toc96901919)

[Figure 53: Spark ML Performance and Node Distribution 52](#_Toc96901920)

**List of Tables:**

[Table 1: Feature Description 20](#_Toc96944020)

[Table 2: Unigram Analysis 35](#_Toc96944021)

[Table 3: Most Common Words 37](#_Toc96944022)

[Table 4: Final Description words 38](#_Toc96944023)

[Table 5: Performance Metrics of the Models 43](#_Toc96944024)

[Table 6: Null Value of the features 47](#_Toc96944025)

[Table 7: CDF Values of Actual Time 49](#_Toc96944026)

[Table 8: Feature Extraction 52](#_Toc96944027)

[Table 9: No. of Features after Encoding 53](#_Toc96944028)

[Table 10: Annova Analysis to understand correlation with Target variable 65](#_Toc96944029)

[Table 11: Regression Performance Metrics 69](#_Toc96944030)

[Table 12: Predictive vs Actual 73](#_Toc96944031)

[Table 13: Hyperparameter Jobs Submitted to Spark Cluster 76](#_Toc96944032)

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**SECOND SEMESTER 2020-21**

**DSECLZG628T DISSERTATION**

Dissertation Title : Automatic Group Categorization and Real Time Ticket Resolution Time Prediction

Name of Supervisor : Sandip Das

Name of Student : Ashish Sarkar

ID No. of Student : 2019hc04098

# Abstract

## 1.1 Problem:

When an issue or support ticket drops into the IT helpdesk- L0, first it needs to be processed and assigned right severity. These tickets are then routed to the right group for resolution, according to the department that would be a perfect fit for the label of that ticket. Hence, the ticket needs to be correctly labeled and severity needs to be defined properly, so that there is no waste of time and resources in routing the ticket to the right agent, only adding to the resolution time of that ticket. In the currently scenario this process is manual, and we are seeing lot of efforts and time are being required to do so. If the ticket is assigned incorrectly then it will delay the overall resolution time of the ticket. Other than this, we have seen the accuracy level for correctly tagging the ticket is approximately 50 to 55% which results in lot of time being wasted to channel the ticket to right group, delaying the overall time to resolve the ticket leading to customer dissatisfaction.

High Quality customer satisfaction is the utmost important when service ticket resolution time performance is considered. L0 team understand the ticket and assign the ticket to respective team. The customer who has issued the ticket is not aware of the actual time taken to resolve the ticket. This leads to increase customer dissatisfaction.

## 1.2 Business Objective:

Automatically classify support tickets: - Automatically classify the IT service desk tickets can done through traditional machine learning models. By mining the historical unstructured ticket description and the corresponding label Ticket classifier models can be trained. The trained model can be used to classify the ticket into respective assigned group along with categorizing urgency, impact, category etc. The model training can be successfully done with previous huge amount of SNOW data. There is definitely some advantage on this

* Faster Resolution time
* Improved Productivity
* More customer satisfaction
* Business growth

The real-world service desk ticket data from a large enterprise IT infrastructure is used for our research purpose.

## 1.3 Automatically predictive support tickets resolution time

A predictive model has been proposed which will estimate/real time SLA the time to complete a ticket by leveraging the hidden structure of previous historical SNOW records and the use of machine learning algorithms. After models are trained with customer data, they are applied to new tickets at the time of creation. Thus, users who raise the ticket can make use of an estimated time-to-completion in early stages of the customer interaction and can inform customer/user.

Real time analysis of the models: - A real time analysis of the predictive and classification model by deploying the model and understands the overall time taken to do this and improvise thus.

## 1.4 Benefits:

* Real time SLA Prediction – more accurate information can be given to the user
* High Performance
* Achieve more customer satisfaction
* Saving huge amount of cost and time

## 1.5 Methodology Used in Classification:

**Exploratory Data Analysis:** This will give us an understanding of the distribution of data and whether we have any imbalanced data

Preprocessing of RAW data and masking of PII and PCI information

**Embedding of textual data -** Word Embeddings are required to use a character-based text as an input for a machine learning model by embedding the text into a vector space (vectorization). This converts the textual data into vectors that can then be used as features in Machine Learning algorithms

**Dimensionality Reduction:** After applying the embeddings, the dataset will grow significantly. The number of columns exceeded the number of rows, and hence it was essential to try Dimensionality Reduction methods on the dataset

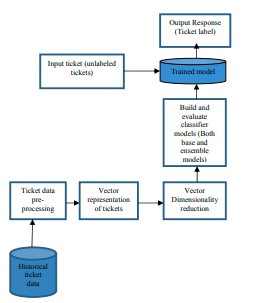
**Model Building** with Naive Bayes, SVM and other classifiers: We will go through several classification model to determine the accuracy of the model

**Model improvement** with Ensemble techniques and LSTM: We will work on increasing the model accuracy by implementing XG-Boost, LSTM and other related models

**Evaluation and comparison** of various ensemble classifiers through different classification metrics like ROC curve, confusion matrix.

**Model Deployment as Microservice:** Once we finalize on the model we will work on effective implementation of the model as microservice

**Future scope:** We will review our model and check what could be the strategy to deploy the model in cloud and add ML-OPS to retrain the model with live data

****

*Figure 1:Model Pipeline*

## 1.6 Potential challenges & risks in doing the project

Proper masking of data is essential as this may expose sensitive data leakage.

Since we are using real data from SNOW it can expose an organization risk because they can lead to adverse consequences and poor decisions making when the model behaves has erroneously in its design or construction, performs poorly or is used inappropriately.

**Key Words**: — Machine learning, Ticket classification, Service desk (Helpdesk), Ensemble classifiers, Naïve Bayes, SVM, XG Boost, LSTM

## 1.7 Methodology Used in Prediction:

The following steps must be understood to know how to build a predictive model.

* The first step is to clean up all the data by eliminating outliers and treating missing data
* Determine whether non-parametric or parametric predictive modeling is more effective
* Reprocess the data into an appropriate format for modeling algorithm
* Specify a subset of the data which is to be used for training of the model
* Train the model parameters to form the trained dataset
* Conduct tests to assess model efficacy
* Validate the accuracy of predictive modeling on the data which is not calibrated
* Send the model for prediction

**Key Words:** — Machine learning, Ticket Time Pre, Service desk (Helpdesk), Ensemble classifiers, Naïve Bayes, SVM, XG Boost, LSTM, Linear Regression, Random Forest Regression, Flask, Light GBM Regression, Cat Boost Regression, Normal Distribution, Cumulative Distribution

**Uniqueness of the project: -**

While ticket classification has already been done earlier in different research fields, this project comprises live real data to classify 180 levels at least. Most of the work earlier has been done with max 10 to 50 labels but complexity of 180 labels and real data has been the most challenging part to get higher accuracy level.

Prediction of real time ticket resolution time based on the past data has never been done earlier. With so much of data and variation on the data, information it has always been challenge in the industry to predict the approximate resolution time other than the SLA mentioned for each ticket based on priority

**Scope & Limitations**

The thesis is limited to building a regression model to successfully learn from past historical data and make predictions about ticket resolution time using only the features that are available at the time of ticket creation. The problem could also be modeled using classification techniques and a comparison drawn between regression and classification results, but no such attempt has been made. However, some work on how we can do the same prediction of real time ticket resolution time using classification has been shown in the appendix section. Finally real time analysis has been, and time analysis has been done by deploying the model in Flask and consuming the model as rest services.

Note: - Most studies have focused on predicting the root cause of the trouble ticket and hence the nature of the problem has been that of classification. Earlier, rule-based learning or expert systems were used to diagnose the trouble tickets automatically [3].

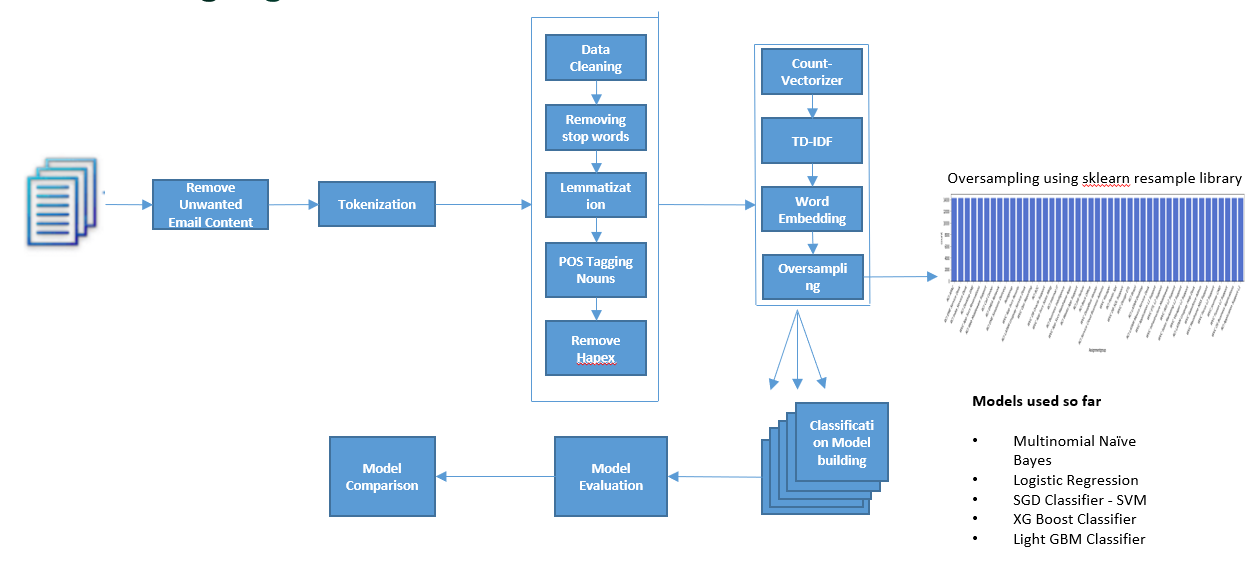
**List of Symbols & Abbreviations used:**

* Term frequency–inverse document frequency 🡪 TF-IDF
* Sequence to sequence 🡪 Seq2seq
* Natural Language Processing 🡪 NLP
* ML 🡪 Machine Learning
* RMSE 🡪 Root Means Square Error
* MAE 🡪 Mean Absolute Error
* NB 🡪 Naïve Bayes

# Introduction

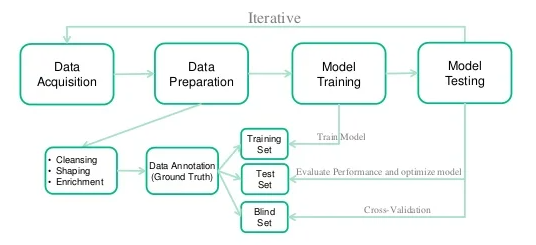
The SNOW dataset considered for the dissertation work had some imbalanced data with some categories containing a greater number of instances and some with a smaller number of instances. Class imbalance problems in a multiclass problem affect the performances of the classifier models.The data set has been collected from SNOW and has 1.9 lakhs of ticket data with 18 feature columns. The user needs to specify the ticket description which is unstructured in nature. The ticket classifier model automatically parses the ticket description - classifies the ticket into one of the correct predefined categories using the user’s ticket description and routes the ticket to proper domain team for resolution. Historical SNOW dataset containing description of ticket and corresponding label, or category is used to train the classifier. Dataset may also contain other structured features but only the feature important description field is taken into consideration for building the ticket classifier model. Target column will be Assignment group. We have used probabilistic model Naïve Bayes, ensemble model XG-Boost, SGD classifier algo and other classification algo to understand how they have performed in terms of accuracy and tried optimizing the performance.

For the predictive modelling we have used the same data with a greater number of features like, time to resolve a ticket, day on which the ticket was created, when it was closed, configuration items, categories, severity, assignment group, SLA etc. The data has been pre-processed. cleaned, EDA done to understand the data, did feature engineering, selection and run through the model to get the approximate time for the resolution of the ticket. Models used so far Linear regression, Ridge regression, Random Forest Regressor, XGB Regressor and Cat-boost regressor.



*Figure 2: Classification Model Pipeline*

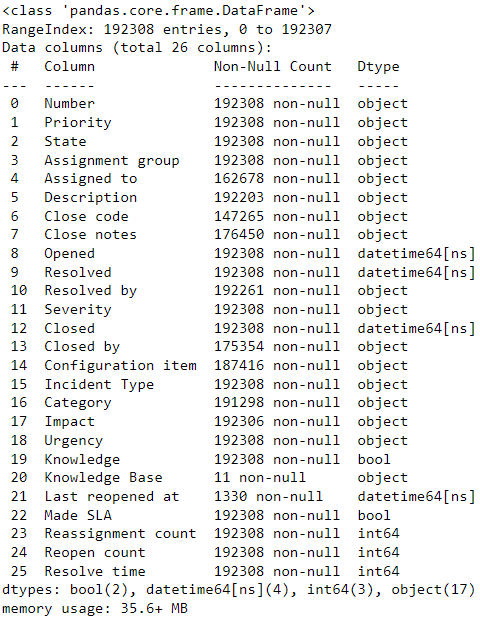
**Model Flow Diagram: Predictive**



*Figure 2:Predictive Model Pipeline*

**Exploratory Data Analysis**

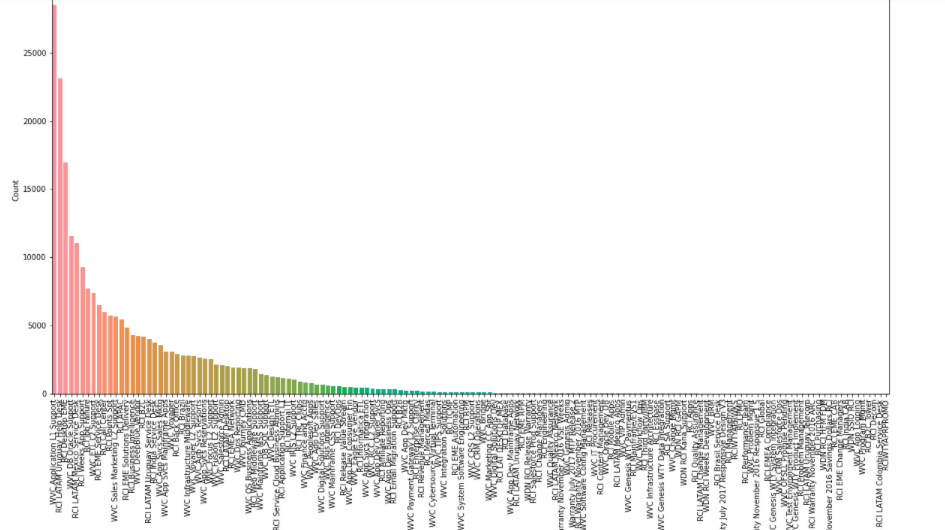
A brief look into the features available in the data set



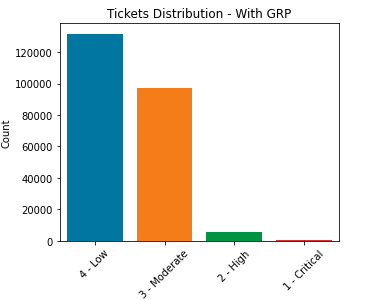
*Table 1: Feature Description*

There are total 4 types of priority. Low, moderate, high, and critical where Low has the highest ticket count and critical has the lowest ticket count. It is clear that there is imbalance data set in this target column. We are considering assigning priority as one of the target columns. we will rename this to Low as 4, Moderate as 3, High as 2 and Critical as 1. For assignment group WVC Application L1 Support group - has the highest number of tickets and RCI Decom has the lowest number of tickets. Groups which have lowest number of tickets needs to cluster into one group and require manual intervention later to assign the tickets into right group. Other than description and Assignment group other columns are not required for the time being. Those has been omitted out.

**Assignment Group distribution**:



*Figure 3:Assigment group*



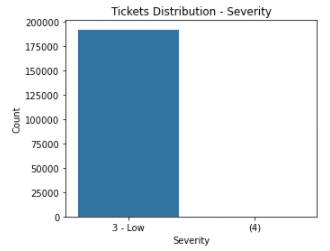
*Figure 4: Ticket Distribution Group*

**Highlights: -**

Target groups: Assignment group, Priority, Category.

For each target group we highly have imbalance data.

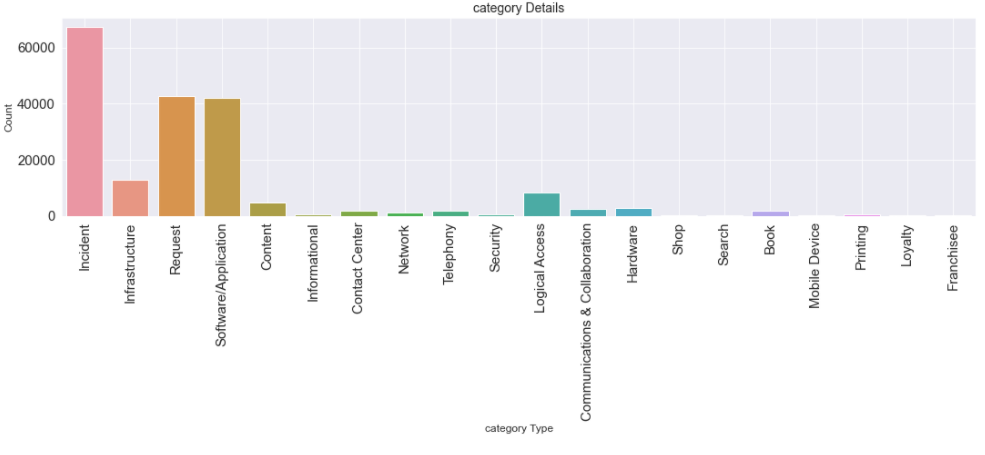
* Severity feature can be removed as it will have no impact 235013 incidents 18 features
* No duplicate tickets
* Critical ticket is very less proving system stability
* Incident is highest category with 85660 number of tickets assigned
* WVC Application L1 Support is the largest Assignment group with 28511 tickets
* Priority 4 are the highest number of tickets



*Figure 5: Severity Bar Graph*

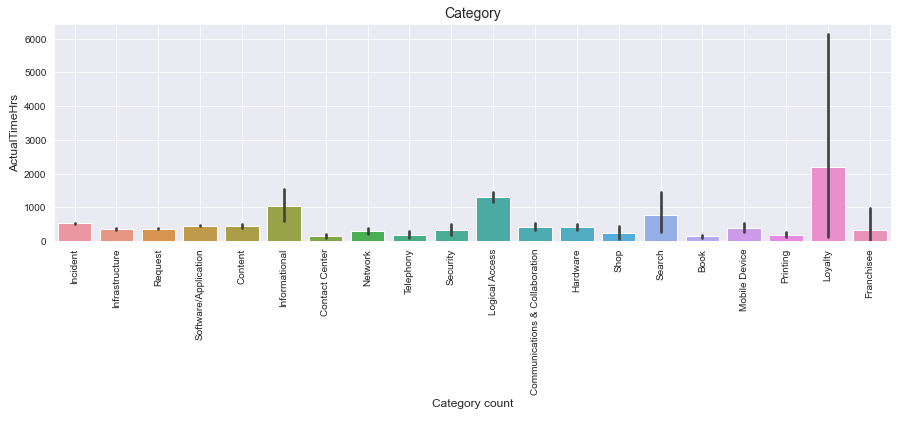
Severity feature can be ignored as all has been tagged to low.

A brief look into category and how category is rated to priority.

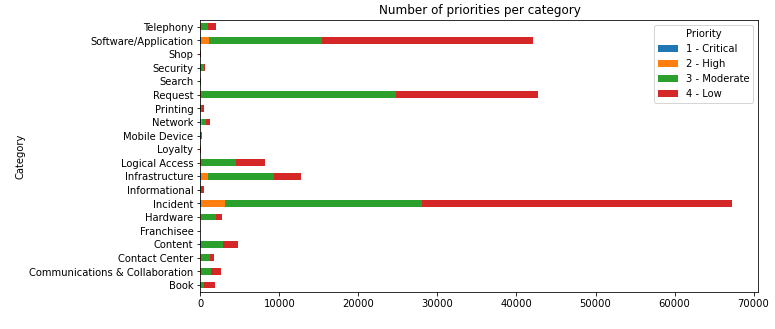


*Figure 6: Category Bar Graph vs Count*

**Category VS Actual Time**



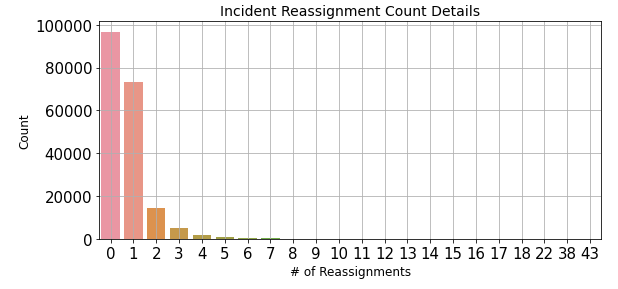
*Figure 7: Category vs Actual Time*



*Figure 8: Priorities per Category*

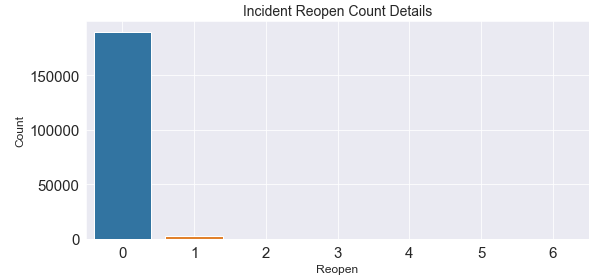
Category feature will play an important role in ticket resolution time prediction.

Loyalty, Logical Access, and Informational category has more ticket resolution time than others. Reassignment count is an important feature. This feature gives an idea on how many times a ticket has been reassigned to the different group. Definitely number of resignment will add to the resolution time of the ticket.

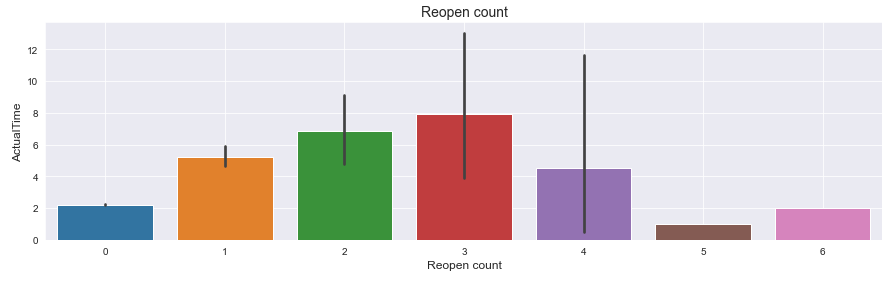


*Figure 9: Reassignment Count*

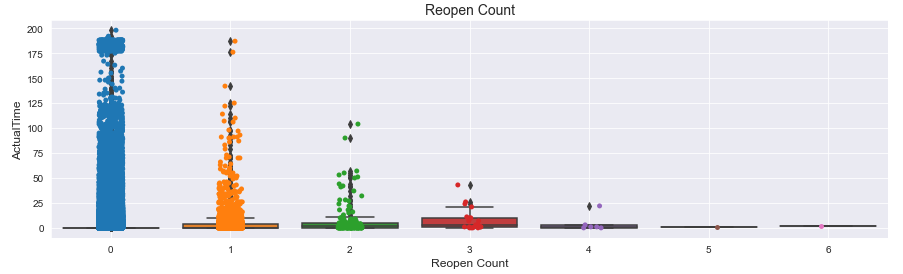
A reopen count can be an important feature. This feature gives an idea on how many times a ticket has been reopened to the different group. Definitely number of resignment will add to the resolution time of the ticket. Most of the tickets has not been reopened except a very few quantities. Hence we can omit this feature.



*Figure 10: Reopen Count*



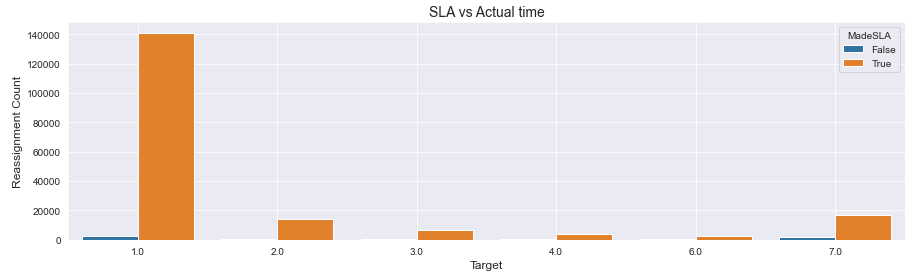
*Figure 11: Reopen Count VS Actual Time*



*Figure 12: Reopen Box Plot Count*

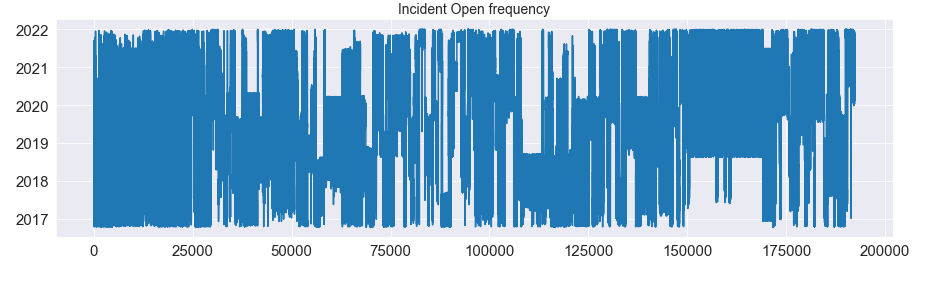
**SLA VS Actual Time**

As seen here the Incidents which failed to maintain the SLA belongs to first category. However, the tickets which take most time to get resolved are the ones breaching very SLA. This is not maintaining industry standard. Hence, we can assume that some of the types of Incidents have more SLA than others.



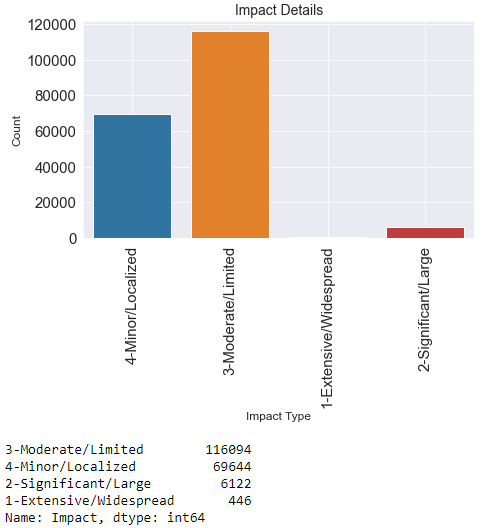
*Figure 13: SLA vs Actual Time*

**Incident open frequency trend**



*Figure 14: Ticket Frequency*

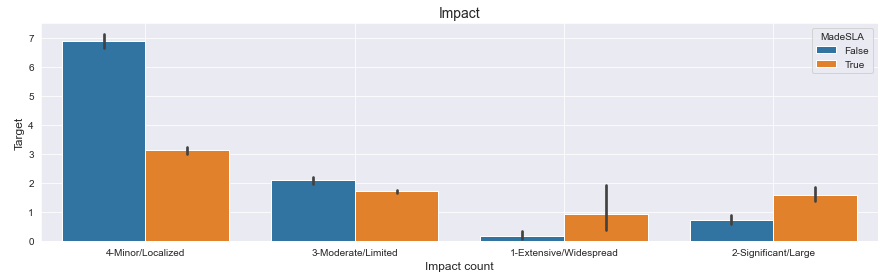
Impact details view and feature will play an important role to find the ticket resolution time. Significant /Large impacted ticket will have more impact in the business. Hence expects lower resolution time. Moderate to be followed next and Minor to be least. There is no extensive / widespread demonstrating the stability of the system.



*Figure 15: Impact Details*

**Impact Vs Actual Time**

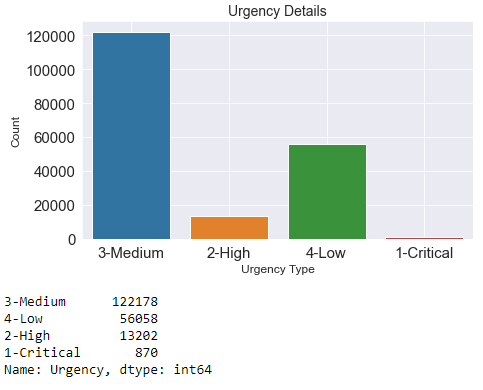
It is clearly visible that significant and Extensive tickets are being resolved at a lesser time than minor and moderate time.



*Figure 16: Impact Details vs Actual Time*

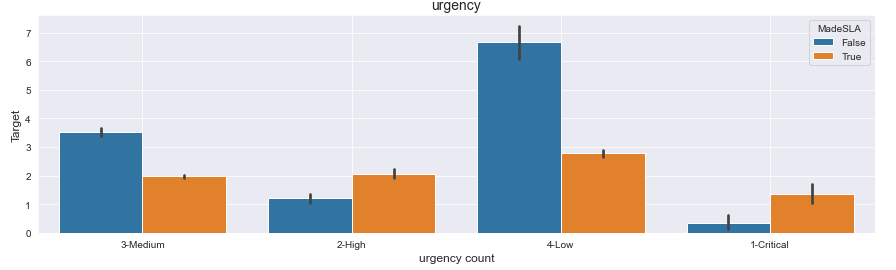
Business Impact has larger impact on resolution.

**Urgency details**



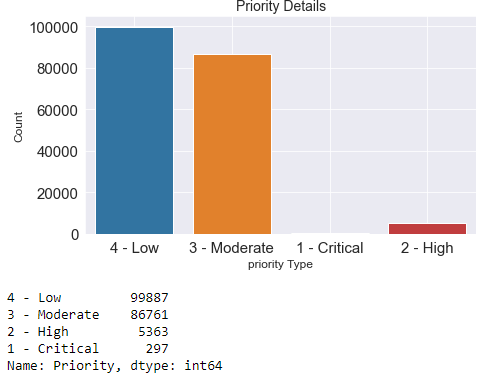
*Figure 17: Urgency*

**Urgency Vs Time**



*Figure 18: Urgency vs Actual Time*

**Priority**



*Figure 19: Priority vs Time*

High and critical business tickets have a faster resolution time rather than moderate and low. This signifies the impact of this feature on resolution time.

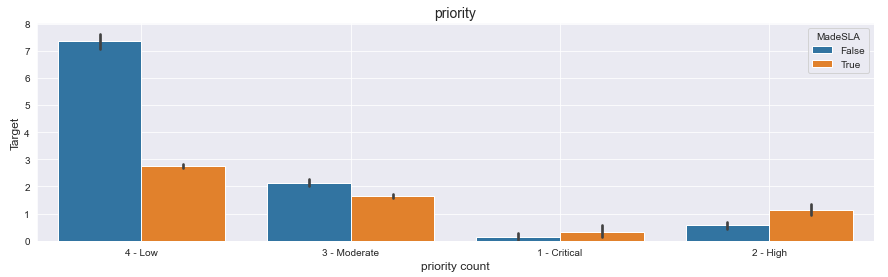
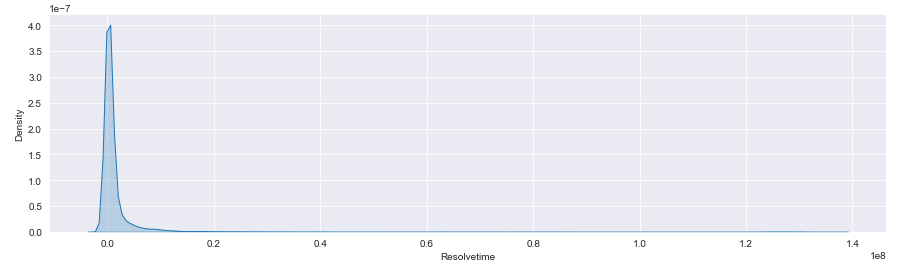


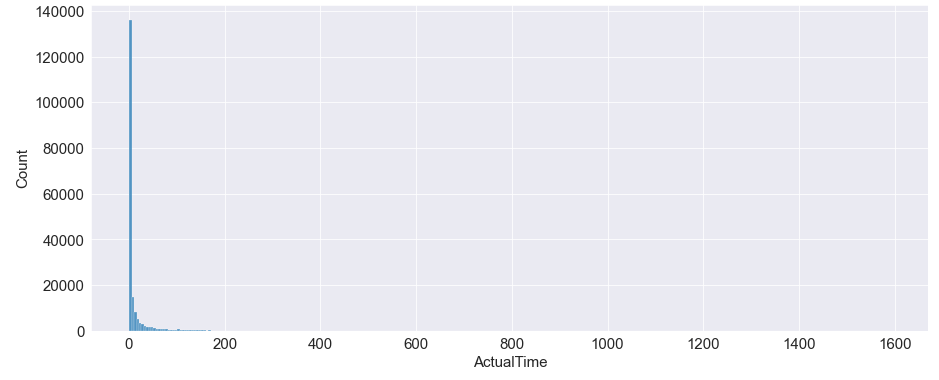
Figure 20: Priory vs Actual Time

**Actual Time**

High Right skewed actual time. There are high chances it has outlier . However maximum ticket has been resolved in a very short time but fewer ticket will have high resolution time



*Figure 21:Distribution of Resolved Time*



*Figure 22: Distribution of Actual Time*

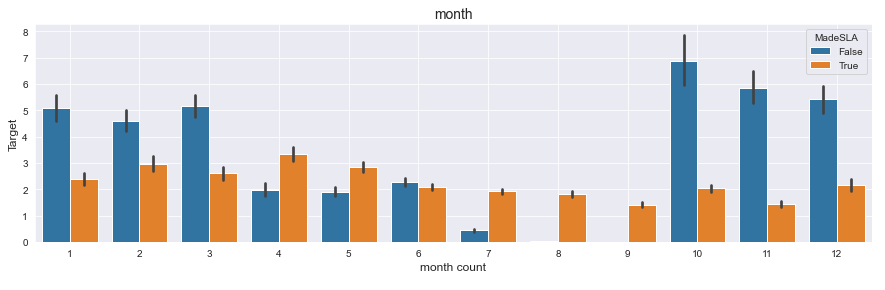
Box plot details of reassignment count vs target. Data is quite distributed and linear regression may not give a better result.

Chart, scatter chart

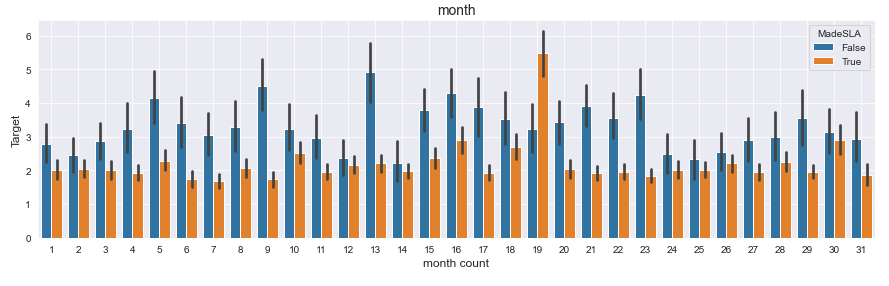
Description automatically generated

*Figure 23: Box Plot of Reassignment Count*

Month vs Target data gives an understanding that from September to March is a period where SLA breaching is more. We will create a feature variable to understand how it is getting impacted.

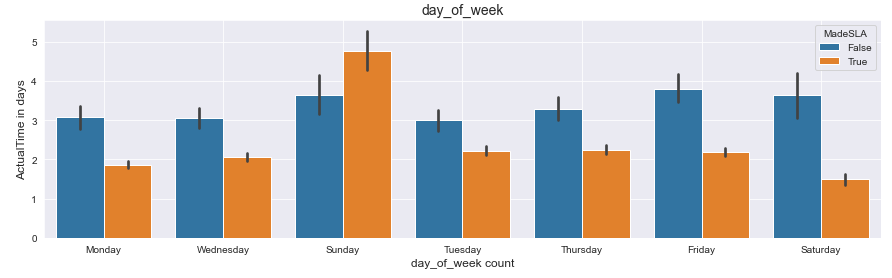


*Figure 24: Month vs Target vs SLA*

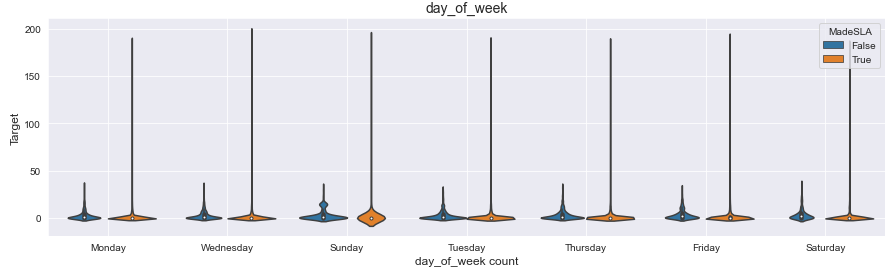


*Figure 25: Month Count vs Target*

Day of week vs time indicates Sunday being a holiday is taking more time than any other days of the week. We will add a feature where we will give more weightage to the tickets which will open on Sunday.



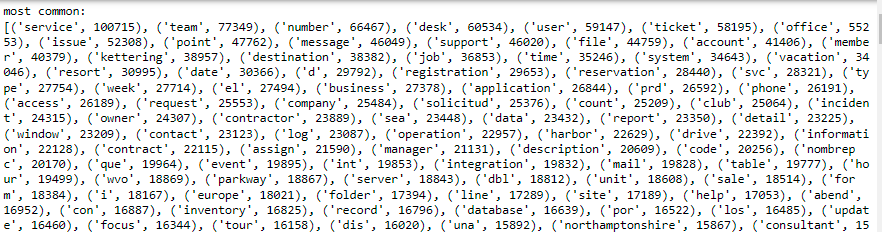
*Figure 26: Day of week vs Target vs SLA*



*Figure 27: day of week Violin Plot*

**Brief understanding of the words in description: -**

We have used NLTK libraries to find the most common unigram words within the description.



**Table 2: Unigram Analysis**

**Unigram Bag of words**: -

Proper structure of the sentence and the relationship between the words that are available in the text description can get the meaning of any sentence. In NLP, the most basic models are based on the Bag of Words (Bow) adding N-gram models like Unigram, Bigram approach or technique, but such models fail to capture the structure the inner meaning of the sentences and the syntactic relations between words. To overcome this issue, POS Tagging has been given more precedence in addition chunking in NLP.

****

*Figure 28: Word Cloud of Unique words*

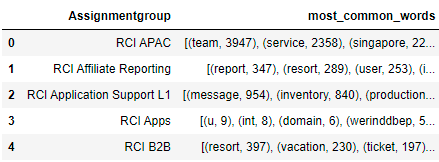
A brief understanding of unique/most common words in the description of WVC DB Support

We can get a clear idea on those issues related to ‘oracle’, ‘rule’, ‘oem’ etc. are key words to identify whether the ticket is assigned to WVC DB support.



*Figure 29: Word Cloud of Oracle Support Group*

A brief understanding of most common words per group

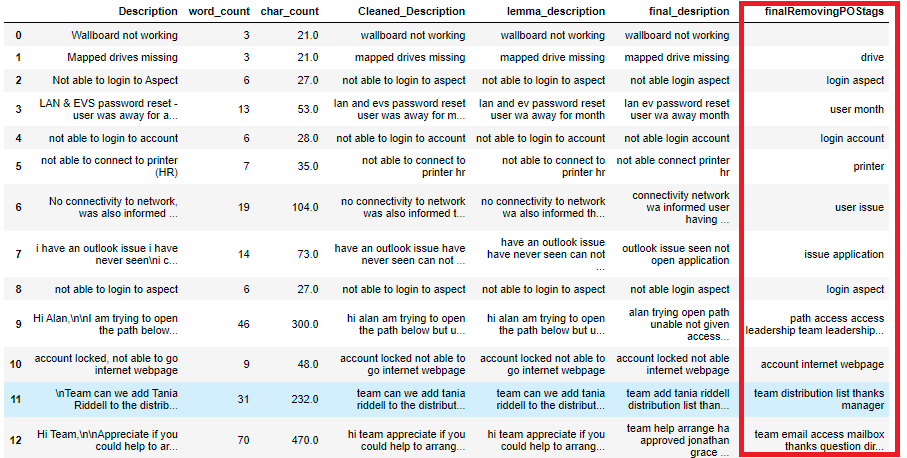


*Table 3: Most Common Words*

**Data Pre-Processing – Classification: -**

The pre-processing of training data is one of the important steps in the data mining process. Pre-processing usually involves cleaning of the raw data i.e., in our case it is the cleaning of the historical ticket data. The dataset should be clean without any noisy and unwanted data in order to build accurate machine learning models. The ticket data used in this work had lot of unwanted data such as names, email address, phone numbers, special characters, date, time, etc. The pre-processing block cleans all such undesired data since they do not help in ticket classification. Also improves the performance of the model. Commonly used stop words and functional words are also removed as a part of pre-processing.

Libraries like Spacy and NLTK has been used for data processing. We saved the pre-processed data set as a pickle file forfurther observation.



*Table 4: Final Description words*

Final description and different removal process. We have considered only lemmatized noun for POS tagging.

## 1.8. Feature Vector representation of ticket data:

Historical ticket descriptions (training data) must be represented in numerical form before applying any machine learning algorithms. A Feature vector is constructed on each ticket description using Term Frequency-Inverse document frequency (TF-IDF) weighting scheme. There are other feature vector representation techniques like Bert , word2VEC however we have seen TD-IDF has better performance in terms of accuracy .In this representation, each element in the vector specifies the unique words taken from the entire corpus and is assigned a TF-IDF value.

## 1.9 Building Classification models: -

In this work, popular probabilistic model like Multinomial Naïve Bayes and most popular boosting algorithm XG-Boost is used to build ticket classifiers. The performance of both base classifier models and boosting classifier models is evaluated and compared using various evaluation metrics such as Accuracy, Precision, Recall and F1-score. At the end of this phase, the model which performs best is chosen and is stored in the model store for the classification of new ticket instance.

We haven’t used

## 1.10. Implementation and experimental results:

### **1.10.1** **Data set analysis and pre-processing**

For the implementation purpose of this research topic, we used the Python as programming language with all necessary libraries such as pandas, scikit learn, matplotlib etc.

Initial analysis of the data revealed the following details.

* Total number of tickets collected: 235013 instances.
* No of distinct classes present in the dataset: 18 classes (hence its multi-class problem).

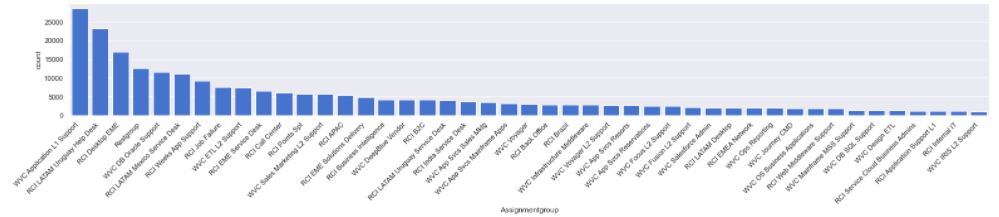
Careful data analysis also reveals that, huge amount of unwanted text, incorrect labelled data (noisy label) and imbalanced classes were present in the raw data. Details of the methods used for pre-processing of ticket has been described in the below section.

* Eliminate duplicates
* Dropping less important features
* Removing punctuation, special characters, short words, emails, number, URLs,
* Non – dictionary words, unreadable characters any hashtag text
* Removing stop words and any other POS except nouns and lemmatized it

We have also understood the most common words both at unigram and bigram level through word clouds.

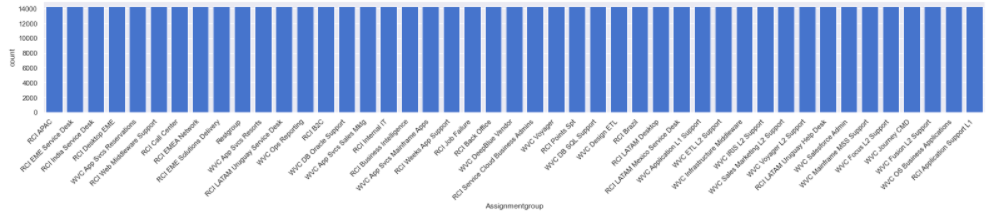
### **1.10.2 Fixing Imbalanced Data Set**

As shared earlier data is on highly imbalanced state - we have worked on the imbalance data set to convert into balanced data set. Oversampling techniques has been used here.



*Figure 30: Imbalance Data Set*

Various techniques can used but we have used sklearn imbalance techniques. Later we will try with SMOTE to understand if we can some performance improvement.



*Figure 31: Balanced Data Set*

## 1.11. Feature Vector representation of ticket data

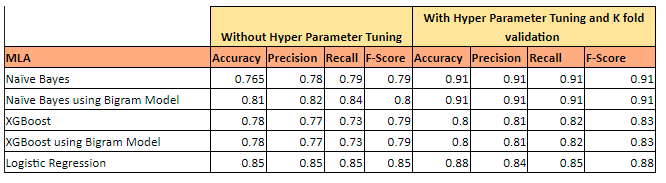
After the pre-processing of the ticket data, numerical representation of the ticket data is done using the vector space model – TD-IDF. We will be using Word2vector model to retrieve digging into documents and identifying content and subsets of content. We will also be reporting the performance of the models in comparison to TD-IDF and Word2Vec.

*Data set has been divided into 80:20 where 80% has been data has been divided into training data set and 20% into test data set.*

## 1.12. Building and Evaluation of the models

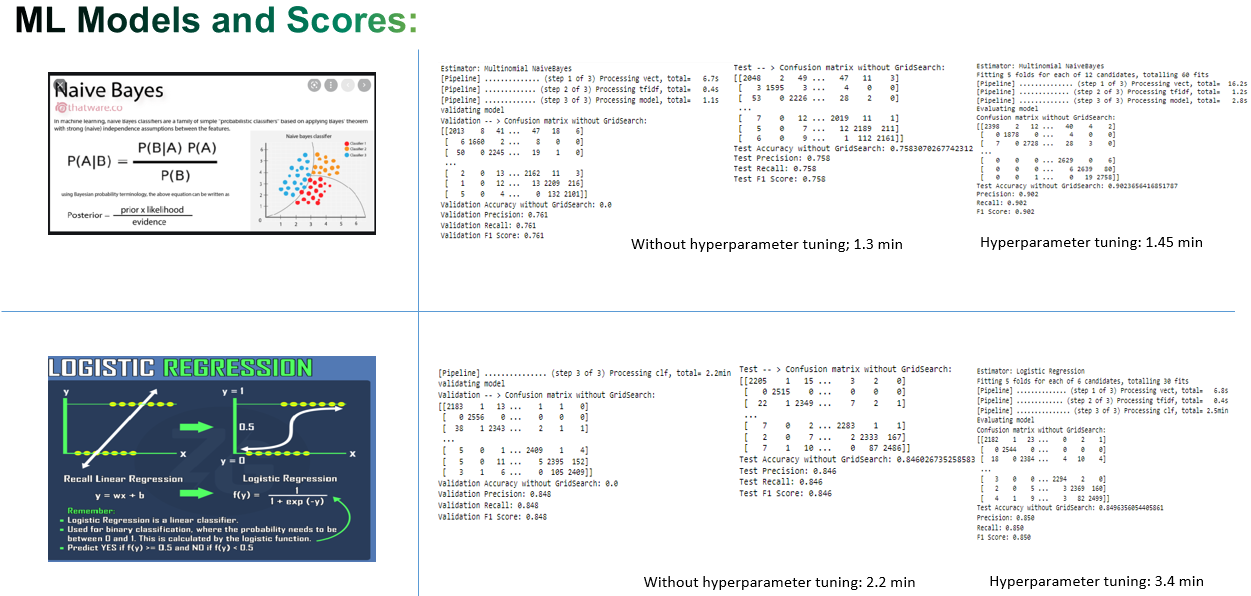
For this research work we have used couple of models so far – Multinomial naïve Bayes, Logistic Regression, XG Boost so far. For hyper parameter tuning we have used grid search CV with 10-fold cross validation. To assess the performance of the model we’ll use two metrics: accuracy, and the F1 score, along with a classification report. The classification report shows that the models perform well. For a quick first attempt we get decent results, with .91% accuracy, and an F1 score of 91.48. With some further tweaks and tuning it should be possible to make it even better.

We will use naïve Bayes as the base model and will try to improve the performance of the model henceforth.

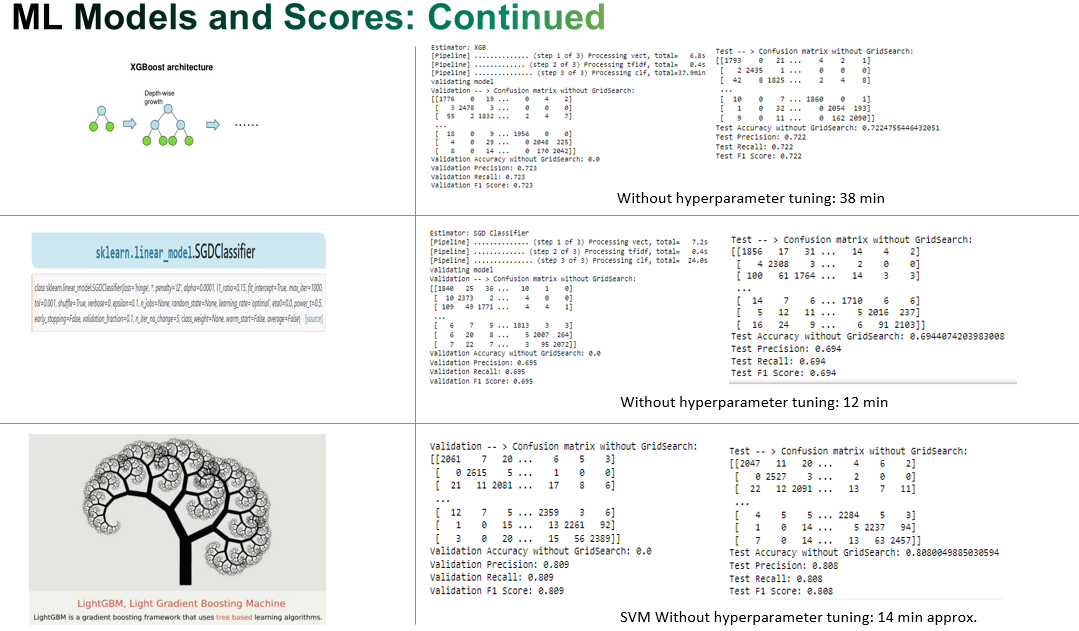


*Table 5: Performance Metrics of the Models*

A brief report on the model performance so far. Multinomial naïve bayes has performed very well with respect to model performance and score. Logistic regression has been the second best however hyper parameter tuning has considerable amount of time. Next plan would be to tweak the models and run the programs in distributed environment to improve the performance of the model.

****

*Figure 32: Model Scores*

****

*Figure 33: Model Scores*

## 1.13. Future Enhancement

The domain specific keywords and stop words could be extracted from the training data to ensure only relevant features will be used for building the models to further improve the accuracy of the system. We plan to explore Deep learning-based classification models with BERT algo , different variation of LSTM to automatically classify the service desk tickets and to investigate its performance on our IT infrastructure ticket data. We will also use libraries which can facilitate the performance of the model.

1.14. Conclusion related to classification models: -

A quality classification model with approximately 90% test accuracy has been built based on supervised machine learning techniques which automatically predicts the category of the ticket using the natural language ticket description entered by user. In this research work, we developed ticket classifier model for one of the real time IT infrastructure service desks taking live data from SNOW. We have used Multinomial Naïve Bayes considering its overall test accuracy coming as 90% and in addition giving a faster performance. The model has been deployed in flask server and the model will be exposed as simple rest service in live deployment. Overall performance of the classification model is approximately 90%.

# PART-B:

While the mid-semester is mainly on the classification model the final semester is more towards

Prediction of actual time a ticket can be taken when it is introduced.

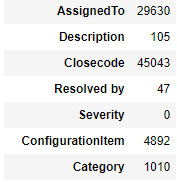
**Prediction of time to be taken to resolve a ticket**

## 1.15. Data Preprocessing:

We have retrieved the data using pandas’ data-frame and did the below data pre-processing steps.

**1.5.1 Null value check: -**

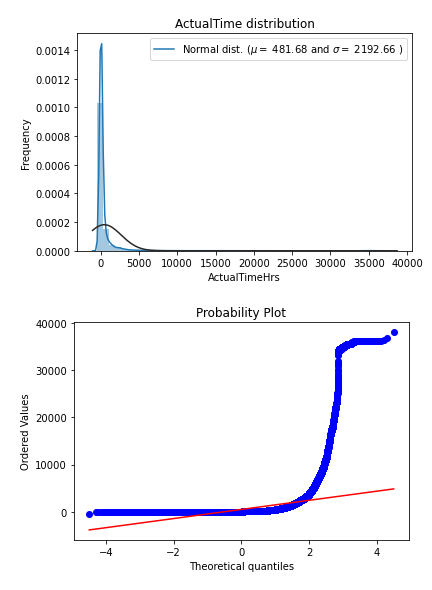
Assigned TO, Description, Closed Code , Resolved BY , Configuration ITEMS and category has null values. We have removed features like assigned to, closed code . Assigned To is not required as we have an important feature – resolved by. we will discuss more on resolved by later. We have removed closed code as this feature is used when a ticket is being closed. We have also removed description as in previous classification model we have already used this feature to generate the features. Resolved BY has only 47 null values for which we have deleted the rows. Configuration item has 4892 items and for that we have used the previous classification model to fill the related information. Regarding category we have used the same techniques used that for configuration item.



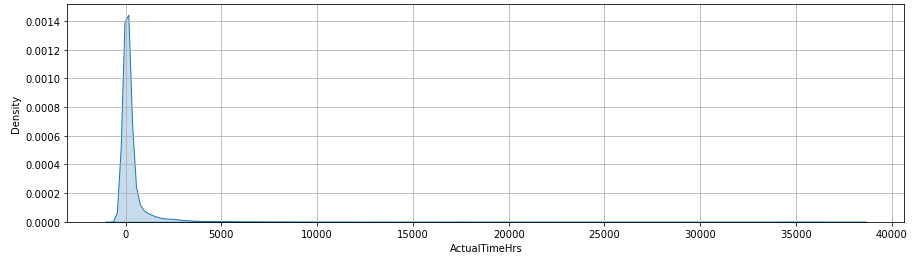
*Table 6: Null Value of the features*

**1.15.2 Removing outlier**

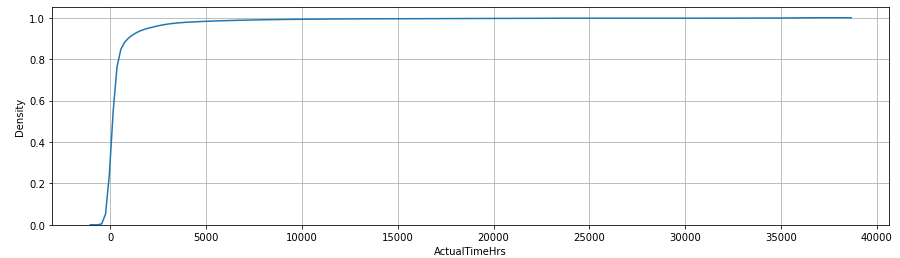
Let’s take a look at the Actual Time distribution of the ticket. To get actual time distribution of the ticket we have used the start date/time of the ticket and closing date/time of the ticket and calculated the actual time by taking the difference in hours.



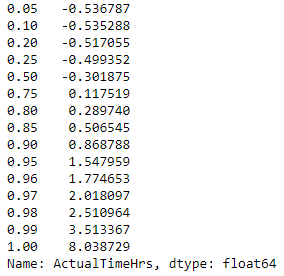
*Figure 34: Actual Time Distribution and QQ Plot*



*Figure 35: Actual Time Distribution*

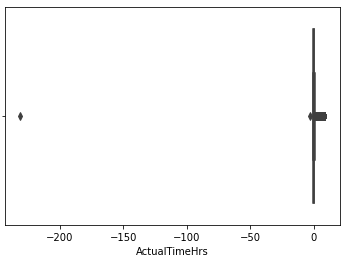


*Figure 36: Actual Time CDF Plot*



*Table 7: CDF Values of Actual Time*

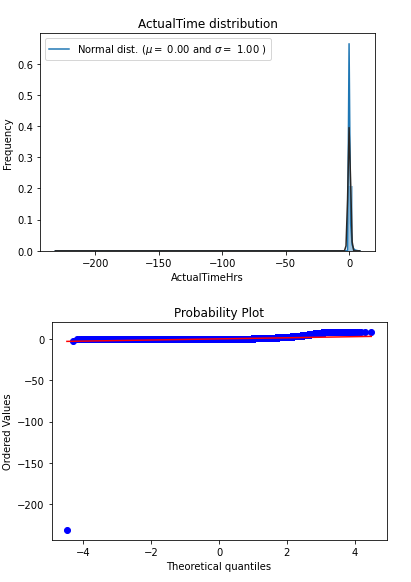
The target variable is right skewed. As (linear) models love normally distributed data, we need to transform this variable and make it more normally distributed. In addition, QQ Plot also gives an idea that target value is not normally distributed. In addition, CDF of Actual time reveals that 99% of the data is within 0.20 value.



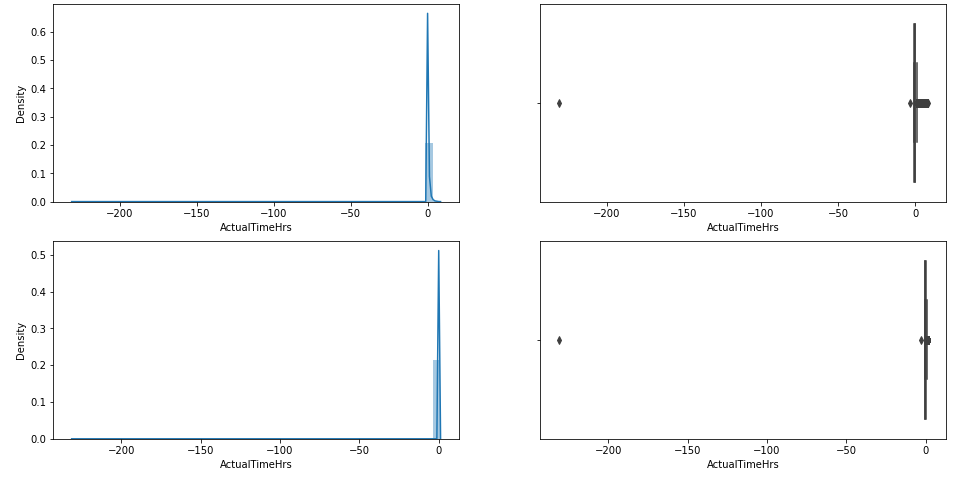
*Figure 37:Actual Time Box Plot after Outlier Removal*

A box plot also reveals that data outlier but not have var.

We have removed the outlier using interquartile range. We can see that its normal distribution and also QQ Plot passes through a straight line depicting normal distribution.



*Figure 38: Actual Time Normally Distributed and QQ Plot*



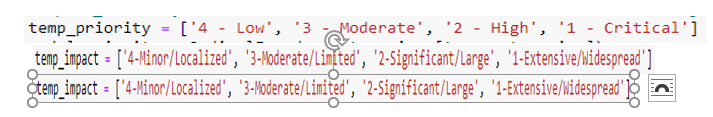
*Figure 39: Comparative study of Actual Time before and after - outlier removal*

## 1.16 Encoding Categorical Values

**1.16.1 Ordinal encoding: -**

Priority, Impact and Urgency has been conserved as the ordinal values.

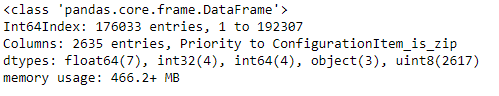
For example priority 1 means high priority ticket issues followed by priority 2, priority 3 and priority 4. Similarly Impact/Urgency can be considered the same way with 1 to be higher side and 4 to be on lower side. SkLearn Ordinal encoding has been used to convert into ordinal values.



*Table 8: Feature Extraction*

**16.2 One Hot encoding: -**

Assignment group, Category, Configuration Item are not ordinal values. We will use sklearn one hot encoding to represent the data in categorical values. However, this has increased the features more than 2500 which will impact the performance of the model. We will feature selection techniques to extract the best features required to built the model.



*Table 9: No. of Features after Encoding*

## 1.17. Improving Machine Learning Models by Adding Features: -

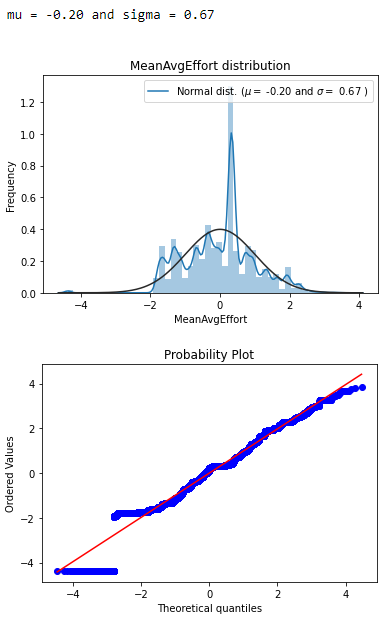
**1.17.1 Start date/time of the ticket: -**

Instead of taking the date/time as a whole we have filter out the date into day of the week, month and date of the month. In EDA we have seen Sunday – day of the week has very important impact on the resolution time. Similarly, month from September to March has higher trend of SLA breaching. Day of the month has also revealed some correlation to the target variable.

We have created a separate feature dow: - to identify whether ticket has been opened on Sunday or not Sunday. This has improved the performance of the model.

Other than dow of the week we have introduced another feature moy -: which identifies whether the ticket has been created from September to March or not. This period is mostly festive period and have positive correlation with target variable

In addition, we have created another variable on tow: - time of ticket which has direct correlation with target variable. Time of creation can be a critical variable for service organizations that work with a non-continuous schedule. For example, extra minutes can be added if a ticket is created during lunch time, or if a ticket is created in a Friday afternoon and has to wait until the next Monday to be processed.



*Figure 40: Mean Average Time*

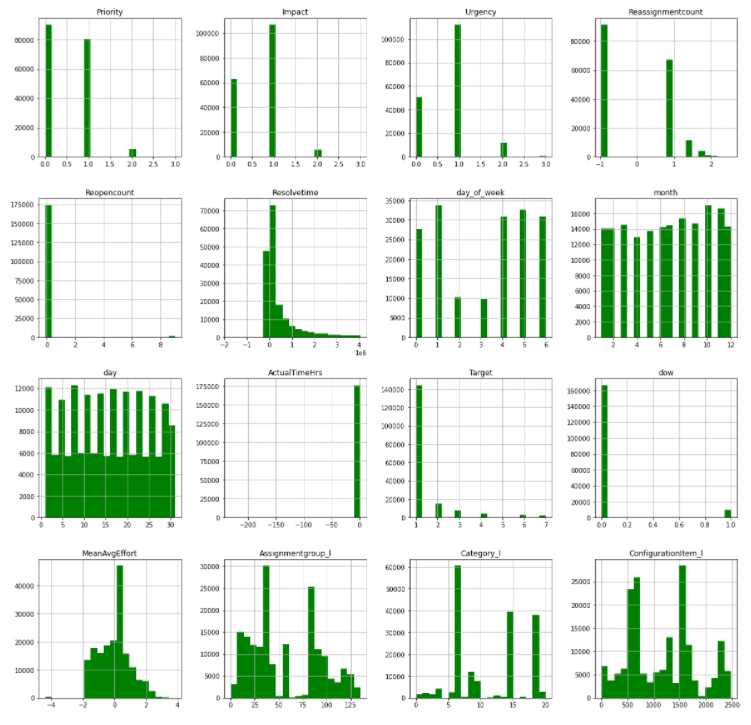
**1.17.2 Performance of the resolved by: -**

Some service agents are more experienced than others in assigning tickets to the right teams. It helps reduce the ticket transition time between teams. We have created a mean average effort continuous variable for each on the associates who has been part of ticket resolution. we have checked both mean, and mode of the variable and seems same.

**Assignment To Resolved By Finally,**

we have also reviewed how assignment to  Resolved BY variable can be correlated to target variable. So if assigned to and resolved by is same then we have seen resolved time is more quick than if different person is assigned to resolve the issue.

**A histogram view of the variable.**

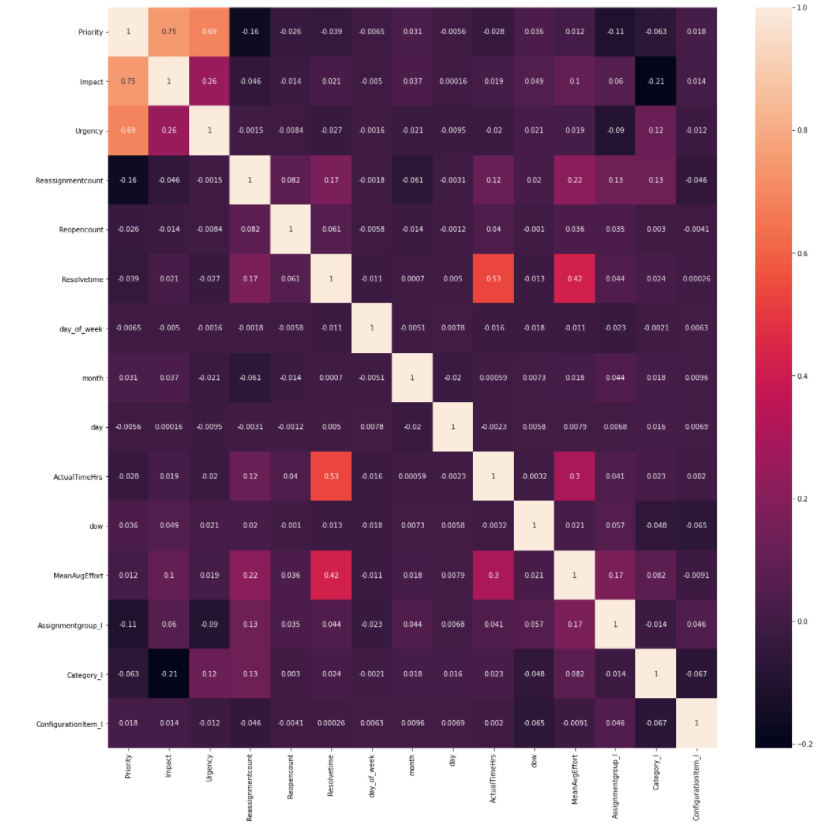


*Figure 41: Histogram Representation of each feature*

**Feature Selection:**

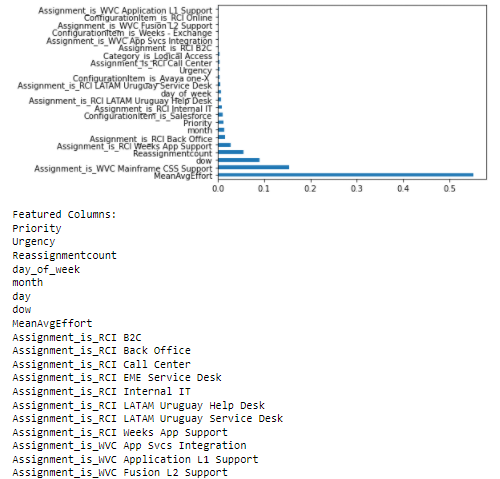
A SNS plot using correlation method to understand whether multicollinearity exists and how they are correlate to the target variable. Priority, Impact and Urgency have some multicollinearity among themselves which can impact on the model accuracy. However, rest looks pretty descent.

Note:- Assigned To , Close code , Knowledge, Made SLA has been dropped off as business wise this will not impact the resolution time of the ticket.

******

*Figure 42: SNS Plot of feature variable vs Target*

Using extra classifier has helped to identify top 30 features out of 2096 feature variable which has surely improved the performance.



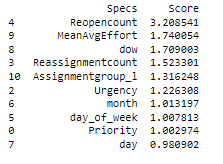
*Figure 43: ExtraTreeclassifier Feature Selection*

A picture containing icon

Description automatically generated

*Figure 44: Random Forest Classification*

ANNOVA analysis and respective score supporting the above statement. It gives clearer indication one how each feature variable is corelated to the target variable on ascending order. Reopen count, mean average effort has major impact on the target variable.



*Table 10: Annova Analysis to understand correlation with Target variable*

**Feature Transformation:** -

We have used both the features in the model and found improved performance but a delta change in the improvement of the accuracy of the model.

Once we have finalized on the featured variable, we have used standard scaler transformation as part of the feature transformation technique.

Input variables may have different units (e.g., feet, kilometers, and hours) that, in turn, may mean the variables have different scales. Differences in the scales across input variables may increase the difficulty of the problem being modeled. An example of this is that large input values (e.g., a spread of hundreds or thousands of units) can result in a model that learns large weight values. A model with large weight values is often unstable, meaning that it may suffer from poor performance during learning and sensitivity to input values resulting in higher generalization error. For example, algorithms that fit a model that use a weighted sum of input variables are affected, such as linear regression, logistic regression, and artificial neural networks (deep learning).

**Train Test Split: -**

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

*Train Dataset: Used to fit the machine learning model.*

*Test Dataset: Used to evaluate the fit machine learning model.*

*We have divided the data into Train: 80%, Test: 20%.*

**Evaluation Metrics for Your Regression Model**

It is necessary to obtain the accuracy on training data, but it is also important to get a genuine and approximate result on unseen data otherwise Model is of no use.

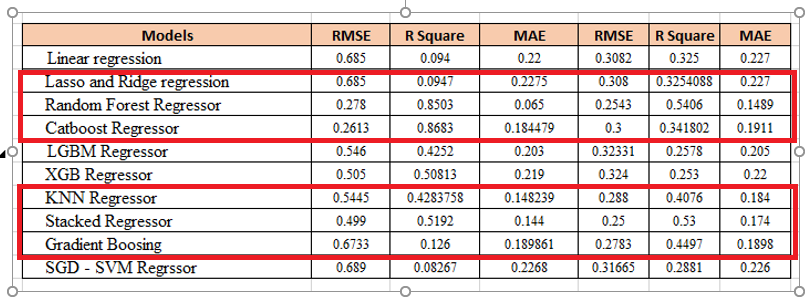
We have used the following important regression metrics to understand the model performance

* MAE: Mean Absolute Error
* Root Mean Squared Error (RMSE)
* R Squared
* Adjusted R Squared

**Model Performance and graphs**

We have used the below models to check how well it works on the accuracy of the models. In addition we have used hyper parameter techniques like Grid Search, Random search CV to improve the model performance such that it can work on the best parameters

* Linear regression
* Lasso and Ridge regression
* Random Forest Regressor
* Catboost Regressor
* LGBM Regressor
* XGB Regressor
* KNN Regressor
* Stacked Regressor



*Table 11: Regression Performance Metrics*

Highlighted models deserve special mention in terms of the better evaluation metrics. Let’s understand each one and the reason behind choosing a model than other.

Based on the performance of the Linear regression we can assume the data is not linear. To be more precise relation between feature variable and predictive variable doesn’t hold linear relationship which the basic assumption for using linear regression.

As we can see Ridge regression has train RMSE 0.68 and test RMSE 0.30. So, we can ignore this regression model. Now we will check few ensembles model both bagging and boosting to check if the model works well here.

Gradient Boosting has RMSE Value train and 0.67 and 0.27 has definitely shown a better performance that the linear models. However, R-Square - how well the data fit the regression model is not that good enough to select it for the model. This is to be followed by some improvement with KNN-Regressor.

As we narrow down the models, it’s a contest between Random Forest, Cat boost Regressor and Stacked Regressor. Considering the data we have Random forest and Stacked Regressor has shown better performance with respect to RMSE value, R-Square and MAE. Random forest

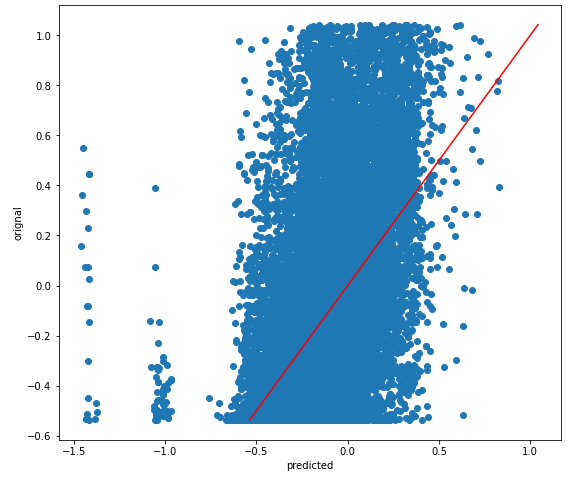
to be the best of all the models has been used close to 0.25, R2 to be 0.51 and finally 0.19. Overall performance of the model has also been good.

Dataset contains features some of which are Categorical Variables and some of the others are continuous variable Decision Tree is better than Linear Regression, since Trees can accurately divide the data based on Categorical Variables.

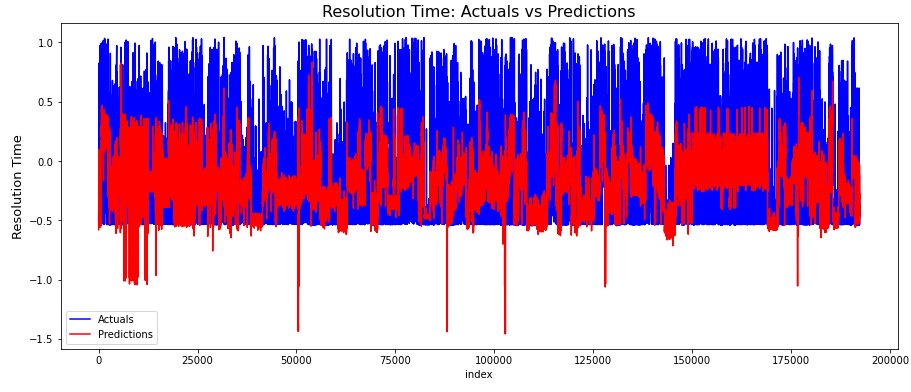
Let’s understand these facts

## 23.1 Linear Regression:

As the linear curve can say that it’s a non-linear data with more spread and of high complexity.



*Figure 45: Linear Regression Plot*

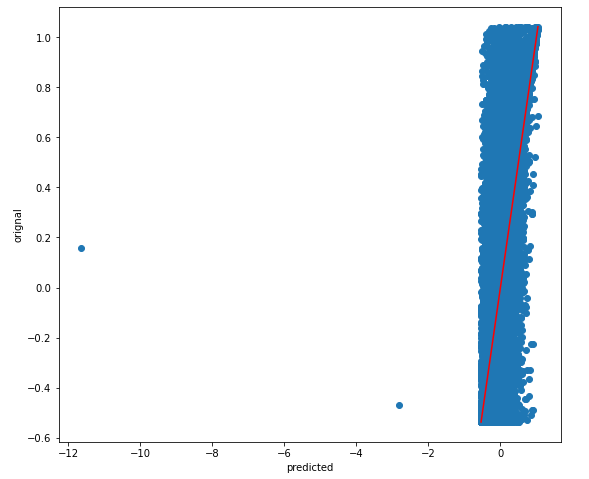


*Figure 46: Actual vs Prediction*

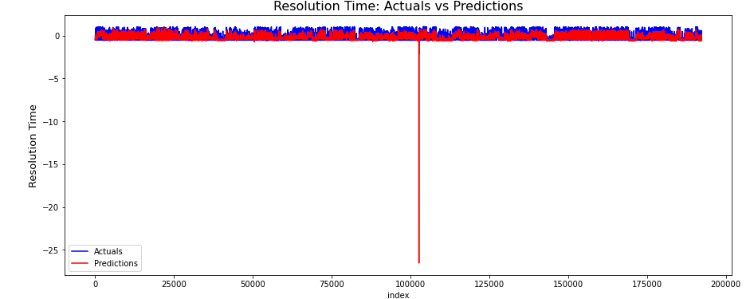
The accrual vs predicted time interval graph shows a good difference between them. Hence it can be said the data is not linear and rather decision tree or ensemble model will have better result.

## 23.2 Random Forest: -

The random Forest regressor has been a better performer and is able to cover most of the data with standard RMSE score. The lower the score its better. We have skipped stacked regression considering overall performance is slow.

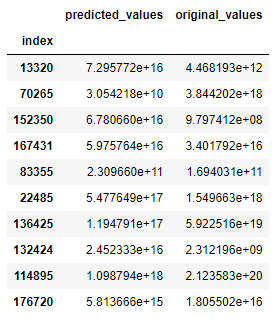


*Figure 47: RF Regression Line*



*Figure 48: Actual vs Prediction plot for RF*

The Actual vs Prediction time is quite close which signifies why the RMSE score is low along with MAE. A brief comparative looks the predicted value vs original value. More or less the expected result is good to go.



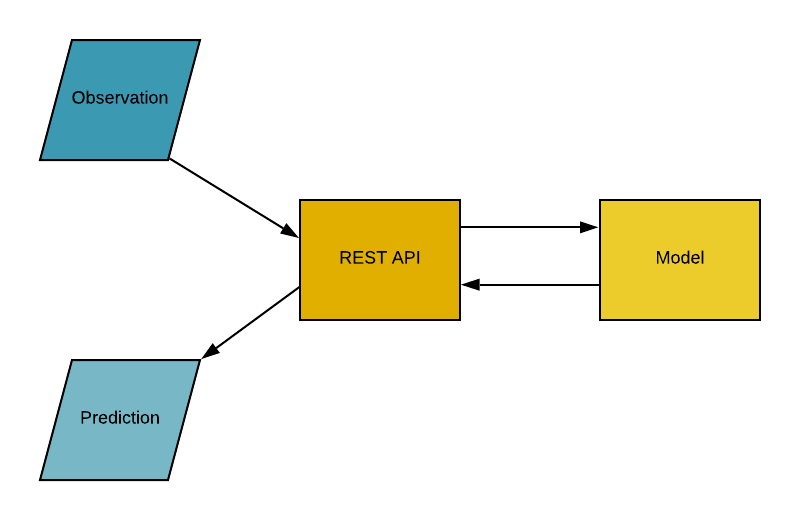
*Table 12: Predictive vs Actual*

24. Real time analysis of the performance of the model: -

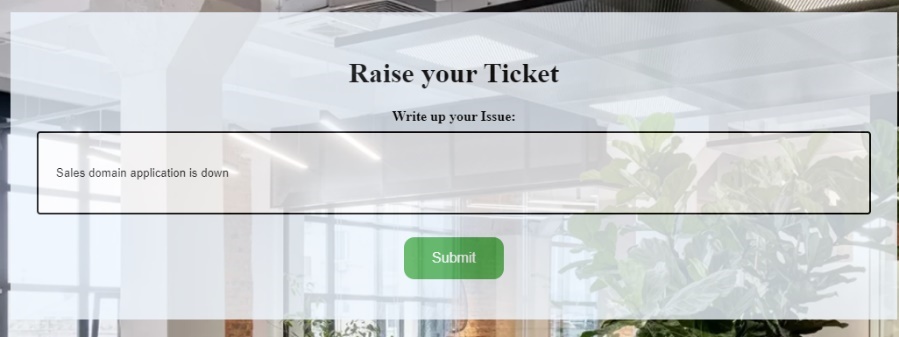
As from the above explanation for the classification of a tickets and assigning into different group, in addition classifying the priority of the model, configuration and category, NAÏVE Bayes model has performed very well w.r.t to other models. Deep neural network model hasn’t been used so far.

Related to understand the real time SLA of the ticket that has been generated random Forest Regression Model has given a solid performance.

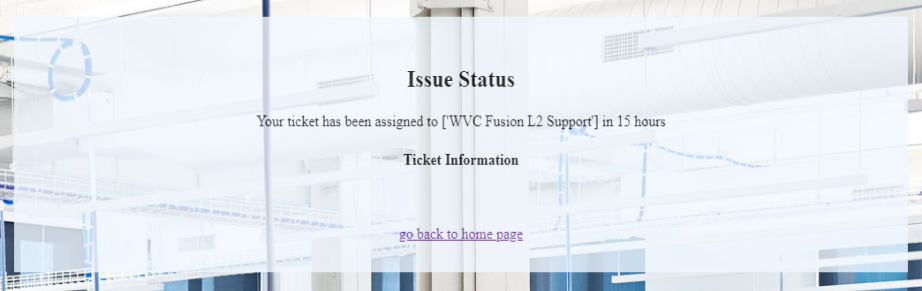
Flask and other python libraries have been used to deploy the models and each model has been exposed as rest services. Once a user type the real issue on a text areas and hits the submit button , the models will automatically classify the different required features from the text explanation and will assign the group and will find the required SLA and share the information with the user in real time. Approximate time to be taken 6 to 7 second.



*Figure 49: Deployed Model Architecture*



*Figure 50: UI Screen 1*

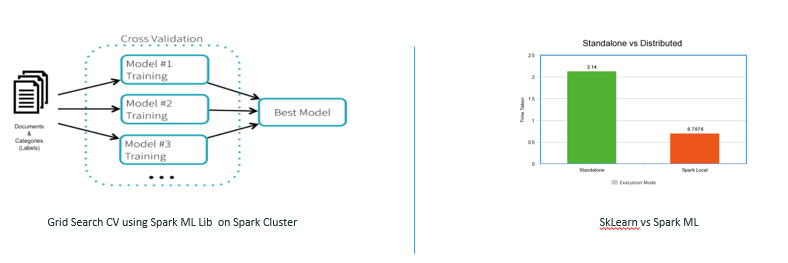
**

*Figure 51: UI Screen 2*

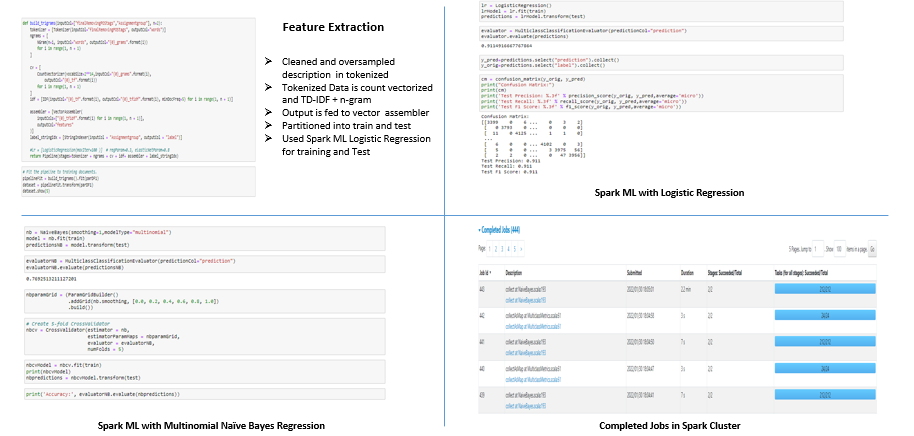
**Performance Improvement of the models:**

Some effort in the research has been given to have a clustered environment – Sparks MLIB

Has shown high prospect of overall improvement in training. The hyper parameter tuning



*Figure 52: Spark ML Performance and Node Distribution*



*Table 13: Hyperparameter Jobs Submitted to Spark Cluster*

# Conclusion:

Based on the classification models used to classify the features of a ticket depending of the description given by the user Naïve Bayes has shown promising performance with respect to speed and accuracy of the model. For getting real time SLA of the ticket linear regression didn’t work well considering non-linearity and spread of the data. In addition, outlier has also impacted the linear regression. In addition, feature columns re a mix of categorical and continuous data. Hence entropy/Gini based decision tree are the best fit here. To improve the performance of the model ensemble model – Random Forest Regressor has been used which not only increases the performance of the model, but overall accuracy has been improved. The other regression technique that worked well is stacked regression. By using the linear combinations of XG Boost, Random Forest and Cat boost Algo and combine the best of the predictions to get the desired output. However, though the overall RMSE value a has decreased by 1 factor overall performance of training the model is lower than Randon Forest.

# Future Enhancement:

Overall requirement can enhance and get better accuracy with Deep learning models. we can use various deep learning regression versions to increase the accuracy of the model. In addition, the text description influence can be checked to understand of the accuracy of the model increases. Once we have a stable model, the training of the model can be taken care in cloud cluster which will increase the efficiency of the model. Trend of the ticket volume can also be analyzed from the data and a new feature can be added as trend based on which accuracy of the prediction can be determined. Finally overall model can be deployed in cloud framework may be Azure or AWS to achieve better performance.

# Bibliography / References:

# *Support ticket classification: - https://towardsdatascience.com/it-support-ticket-classification-and-deployment-using-machine-learning-and-aws-lambda-8ef8b82643b6*

# *Understand Ticket Classification https://www.analyticsinsight.net/it-ticket-classification/*

# *Cloud deployment of Model https://towardsdatascience.com/it-support-ticket-classification-and-deployment-using-machine-learning-and-aws-lambda-8ef8b82643b6*

# *Article on Classification* [*https://www.sciencedirect.com/science/article/pii/S1319157819300515*](https://www.sciencedirect.com/science/article/pii/S1319157819300515)

# *Research Work Ticket Classification* [*https://arxiv.org/ftp/arxiv/papers/2103/2103.15822.pdf*](https://arxiv.org/ftp/arxiv/papers/2103/2103.15822.pdf)

**Machine Learning Flask deployment** [*https://www.analyticsvidhya.com/blog/2020/04/how-to-deploy-machine-learning-model-flask/*](https://www.analyticsvidhya.com/blog/2020/04/how-to-deploy-machine-learning-model-flask/)

Li J, Monroe W, Ritter A, Galley M, Gao J, Jurafsky D. Deep reinforcement learning for dialogue generation. arXiv preprint arXiv:1606.01541. 2016 Jun 5.

Leah S. Larkey and W. Bruce Croft. Combining classifiers in text categorization. In SIGIR-96, 1996.

Khan, B. Baharudin, L.H. Lee, Kh. Khan, A Review of Machine Learning Algorithms for Text-Documents Classification, Journal of Advances in Information Technology, 1(1):4–20, 2010

S. Agarwal, R. Sindhgatta, B. Sengupta, SmartDispatch: enabling efficient ticket dispatch in an IT service environment, ACM SIGKDD (2012) 1393–1401

V. Shimpi, M. Natu, V. Sadaphal, et al., Problem identification by mining trouble tickets, in: Proceedings of the 20th International Conference on Management of Data, CSI, 2014, pp. 76–86.

# Analysis Of Nonlinear Regression Models: https://www.researchgate.net/publication/51422290\_Analysis\_Of\_Nonlinear\_Regression\_Models\_A\_Cautionary\_Note

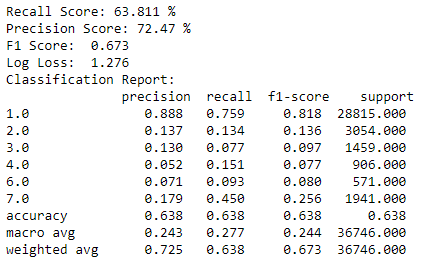
# Appendices:

The other way of find the SLA based resolution time of a ticket.

The idea is to have a near to accurate SLA based model, there are two ways it can done. Once to use the regression model and the other to have a classification model. The research work has also tried to get some idea on how it can be achieved through classification. The best way to is the understand the actual time in hours and bin the same into different categories like 8 hrs. 16 hrs., 20 hrs., 1 day, 2 days etc. We have used binning technique to get the labels done. Once we have the right label in place, data preprocessing, feature transformation has been done. The trained data has been fed into different model to understand the accuracy of the model based on classification metrics.



*Figure 53 Binning Classified Resolution Time*



*Figure 54 Classification Accuracy*

**Check list of items for the Final report:**

Is the Cover page in proper format? Y

Is the Title page in proper format? Y

Is the Certificate from the Supervisor in proper format? Has it been signed? Y

Is Abstract included in the Report? Is it properly written? Y

Does the Table of Contents page include chapter page numbers? Y

Does the Report contain a summary of the literature survey? Y

Are the Pages numbered properly? Y

Are the Figures numbered properly? Y

Are the Tables numbered properly? Y

Are the Captions for the Figures and Tables proper? Y

Are the Appendices numbered? Y

Does the Report have Conclusion / Recommendations of the work? Y

Are References/Bibliography given in the Report? Y

Have the References been cited in the Report? Y

Is the citation of References / Bibliography in proper format? Y

A picture containing text

Description automatically generated Text

Description automatically generated with low confidence

(Signature of Student) (Signature of Supervisor)

Date: 02/27/2022 Date:02/28/2021