*A Project Report On*  
**“Classification of User Support Tickets through Text Analysis using Machine Learning”**

*Submitted in partial fulfilment of the*  
*requirement for the Award of degree of*  
**Bachelor of Technology**

in

**Computer Science Engineering**  
*SRM Institute of Technology*

**Submitted by:**  
Shraman Das

**Under the guidance of:**  
Mr. Vivek Gaurav Saini

(Sr. Programming Officer)

GEOPIC, ONGC, Dehradun



**DECLARATION**

I hereby declare that the work presented in this project report entitled  
**“Classification of User Support Tickets through Text Analysis using Machine Learning”** in partial fulfilment of requirement for the Award of Degree of Bachelor of Technology, submitted in the department of Computer Science Engineering, SRM Institute of Technology, Chennai, is an authentic record of my work carried out during the Industrial Training from **03 June 2019 to 03 July 2019**, under the guidance of **Mr.Vivek Gaurav Saini**, GEOPIC, ONGC, Dehradun.

**Signature of the Student**  
Shraman Das

**CERTIFICATE**

This is to certify that **Mr. Shraman Das** a student of B.Tech. (CSE) SRM Institute of Technology, has done his Summer Training at **GEOPIC, ONGC Dehradun.** The project work entitled **“Classification of User Support Tickets through Text Analysis using Machine Learning”** embodies the original work done by him during the period 03 June 2019 – 03 July 2019.

**Signature of Project Guide Signature of Training Coordinator**

**(Mr. Vivek Gaurav Saini) (Mr. P.R.Meena)**

Sr. Programming Officer D.G.M. (Programming)

**ACKNOWLEDGEMENT**

The summer training at ONGC is a great opportunity for learning and self- development. I am honoured to be a part of it and have such a wonderful experience working there. The guidance provided was equally wonderful. I am grateful to my mentors for helping me throughout the project at every phase.

It gives me immense pleasure and a sense of satisfaction to have an opportunity to acknowledge and to express gratitude to those who were associated with me during my industrial training at GEOPIC, ONGC Dehradun.

I also express my sincere thanks and gratitude to ONGC authorities for allowing me  
to undergo the training in this prestigious organization. I would like to thank our Project Head, **Mr. V.K Sharma, CGM (Programming)** andProject Guide, **Mr. Vivek Gaurav Saini (Sr. Programming officer)** for providing me the technical guidance, opportunity and infrastructure to work. I am extremely thankful to them for entrusting me with such good guidance and support all along my training.

**ABSTRACT**

Machine learning for [natural language processing](https://www.lexalytics.com/lexablog/what-is-natural-language-processing) and [text analytics](https://www.lexalytics.com/lexablog/text-analytics-functions-explained) involves using machine learning algorithms and “narrow” artificial intelligence (AI) to understand the meaning of text documents. These documents can be just about anything that contains text: social media comments, online reviews, survey responses, even financial, medical, legal and regulatory documents.

In this project we have used two Supervised Learning Algorithm to apply text analysis to train a machine to be able to classify the user issues.

These algorithms are –

* Naive Bayes
* Random Forest

We have used NLTK library which contains various algorithms of text analysis like- tokenization, lemmatization, POS tagging etc. We have a total of 3000 User Cases, out of which 500 cases are labelled and are used to train the machine and the remaining 2500 cases are unlabelled. Using the training acquired by the machine we use it to categorize the unlabelled issues.

The various accuracy and precision parameters like – Precision, Recall, F1 Score and Accuracy Score are evaluated for each algorithm. The result is saved in a file to be used in future to classify a text.

**CONTENTS**

**Title Page No**

Declaration................................................................................................................ii

Certificate.................................................................................................................iii

Acknowledgement...................................................................................................iv

Abstract....................................................................................................................v

Contents ..................................................................................................................vi

**CHAPTER 1 Introduction**

* 1. Problem Statement..........................................................................................1-2
  2. Purpose...............................................................................................................2
  3. Objective.........................................................................................................2-3

**CHAPTER 2 Project Description**

* 1. What is text analysis...........................................................................................4
  2. Data pre-processing.........................................................................................4-5
  3. Bag of words – Vectorization.............................................................................5
  4. Finding TFIDF...................................................................................................5
  5. NLTK library......................................................................................................6
  6. Algorithm 1 – Naive Bayes.............................................................................6-7
  7. Algorithm 2 – Random Forest Classifier.........................................................8-9
  8. Comparison between Naive Bayes and Random Forest Classifier.....................9
  9. Model accuracy and precision........................................................................9-11
  10. Python GUI..................................................................................................11-13

**CHAPTER 3 Coding and output**

3.1 Training and testing.......................................................................................14-21

3.2 GUI................................................................................................................21-25

**CHAPTER 4 Conclusion** ......................................................................................26

**CHAPTER 5 Future scope**...................................................................................27

**CHAPTER 6 Refrences**........................................................................................28

**INTRODUCTION**

* 1. **Problem Statement**

**Background :** The Geodata Processing and Interpretation Centre (GEOPIC) is a premier seismic processing and interpretation WorkCentre of ONGC. The Geoscientists at the WorkCentre work on high end work-stations with specialised seismic processing and interpretation software.

There is a software support team in place which oversees all the system administration, software maintenance and support for the above mentioned software. There is an existing ticketing system wherein the users (Geoscientists) of the system/software register their day-to-day problems, installation requests and other miscellaneous requests, which is then acted upon and resolved by the Software support personnel. The existing Ticketing system does not have any categories to the kind of user request and all the requests are stored and termed as `User Issue`.

The software development team of GEOPIC is about to roll out a new and improved ticketing system wherein the user’s will first select the kind of request. There are X such different kinds of request (‘Issue’, ‘Software Installation’, ‘Project Creation’,…….). The database structure in new application has an additional column for request- Category in the ticket table. There are a total of 9 categories for the requests, namely –

1. Issue
2. Software Installation
3. Project Creation
4. Project Backup
5. Project Restore
6. Tape Read/ Write
7. Data Transfer
8. User Creation / Deletion / Access
9. Printer / Plotter

**Problem :** The user cases from old application need to be migrated from old database to new database, however, since the table in old database do not have any categories assigned to user tickets, it is now a task to identify and classify the old tickets into the newly defined categories so that old cases may also be visible in new system. There are around 3000 such old user cases that need to be categorised.

**Solution Approach :** To devise a supervised machine learning model that analyses text of the User Ticket ‘Title’ and ‘Description’ columns. The learning of the machine is to be facilitated using 500 out of the 3000 cases against which categories have been supplied manually (labelled examples). The machine learning model shall then be able to predict the category of the remaining 2500 cases(unlabelled examples) using the model it has built.

* 1. **Purpose**

Machine Learning is a powerful tool to implement a range of functionalities on the machine. It basically means “teaching the machine”. With such large chunks of data gathering every second we need to get some meaning from data collected.

Text classification is a smart classification of text into categories. And, using machine learning to automate these tasks, just makes the whole process super-fast and efficient. Artificial Intelligence and Machine learning are arguably the most beneficial technologies to have gained momentum in recent times. They are finding applications everywhere.

[**Intent**](https://www.paralleldots.com/intent-analysis)**,**[**emotion**](https://www.paralleldots.com/emotion-detection)**and**[**sentiment analysis**](https://www.paralleldots.com/sentiment-analysis) of textual data are some of the most important parts of text classification. These use cases have made significant buzz among the machine intelligence enthusiasts. We have developed separate classifiers for each such category as their study is a huge topic in itself. Text classifier can operate on a variety of textual datasets. You can train the classifier with tagged data or operate on the raw unstructured text as well. Both of these categories have numerous application of themselves.

* 1. **Objective**

Using text classification to understand text sounds like magical thinking. People constantly create and evolve language. A classification algorithm does not care what language the text is in as long as it can at least break apart the text into separate words and measure the effects of those words. As long as you give the classifier enough training data to cover a wide range of possible English words.

The goal of text classification is to automatically classify the text documents into one or more defined categories. Some examples of text classification are:

* Understanding audience sentiment from social media,
* Detection of spam and non-spam emails,
* Auto tagging of customer queries, and
* Categorization of news articles into defined topics.

The WorkCenter operates on thousands of gigabytes of data. Hence, it is certain that the systems on which the work is done will face some issues and faults. These issues need to be forwarded to the right departments so that they can be resolved. For this the issues arising need to be manually categorized into their respective categories and then resolved. This is a hectic task when we need to do this on a daily basis.

The aim here is to create a machine learning model that will analyse the issue generated, process it and automatically categorize it to its respective category. This way the issue generated can be directly forwarded to designated departments, thus reducing manual cost and effort.

**PROJECT DESCRIPTION**

**2.1 What is text analysis?**

Text Analysis is about parsing texts in order to extract machine-readable facts from them. It is the automated process of obtaining information from text.

The purpose of Text Analysis is to *create structured data* *out of free text content*. The process can be thought of as slicing and dicing heaps of unstructured, heterogeneous documents into easy-to-manage and interpret data pieces.

A text fed into a neural network passes through several stages of analysis. The first is sentence segmentation, in which the software finds the sentence boundaries within the text. The second is tokenization, in which the software finds individual words. In the third stage, parts-of-speech tags are attached to those words, and in the fourth, they are grouped according to their stems or concepts, in a process known as lemmatization. That is, words such as be, been and is will be grouped since they represent the same verb idea.

**2.2 Data Pre-processing**

Data pre-processing is required to convert human language to machine understandable format for further processing. It includes the following steps :

1. **Tokenisation :** It is the process of converting text into tokens. Tokenisation is of two types –

* Sentence tokenisation - It converts a given text into group of sentences. It displays one complete full sentence.
* Word tokenisation – It converts text into separate words. Entire sentence is broken down into its constituent words.

1. **Converting into lower case :** entire text in converted into lower case for uniformity.
2. **Removing punctuation marks :** since punctuations and symbols are not providing any useful information , they are removed.
3. **Removing Stopwords :** Stopwords are the words that frequently occur in English, and do not have any semantics ; like – ‘is’, ‘are’, ‘were’, ‘was’, ‘the’ etc.
4. **Stemming :** It refers to reduction of words to their root by dropping unnecessary characters , especially suffixes(‘-s’, ‘-es’).
5. **Lemmatization :** Reducing the word to its base form. Eg – ‘opening’ is converted to ‘open’ . this is done to ensure that different forms of same word are treated equally.

**2.3 Bag of Words - Vectorization**

Machines, unlike humans, cannot understand the raw text. Machines can only see numbers. Particularly, statistical techniques such as machine learning can only deal with numbers. Therefore, we need to convert our text into numbers.

The most commonly used approach to convert words into numbers is to implement the concept of : Bag of Words(BoW).

**CountVectorizer** class from the **sklearn.feature\_extraction.text** library is used to find BoW. The **CountVectorizer** provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.

**2.4 Finding TFIDF**

The bag of words approach works fine for converting text to numbers. However, it has one drawback. It assigns a score to a word based on its occurrence in a particular document. It doesn't take into account the fact that the word might also be having a high frequency of occurrence in other documents as well. [TFIDF](https://en.wikipedia.org/wiki/Tf%E2%80%93idf) resolves this issue by multiplying the term frequency of a word by the inverse document frequency. The **TF** stands for **"Term Frequency"** while **IDF** stands for **"Inverse Document Frequency".**

* Term-Frequency = (no. of occurrences of a word ) / (total words in document)
* IDF = log((total no. of documents) / (no of documents containing the word))

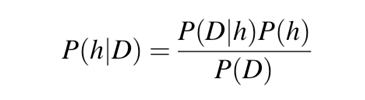
**2.5 NLTK library**

* NLTK stands for Natural Language ToolKit.
* The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP).
* NLTK consists of the most common algorithms such as tokenizing, part-of-speech tagging, stemming, sentiment analysis, topic segmentation, and named entity recognition. NLTK helps the computer to analysis, preprocess, and understand the written text.

**2.6 Algorithm 1 – Naive Bayes**

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets.

Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features. This is a strong assumption but it simplifies computation, and that's why it is considered as naive.



* P(h): the probability of hypothesis h being true (regardless of the data). This is known as the prior probability of h.
* P(D): the probability of the data (regardless of the hypothesis). This is known as the prior probability.
* P(h|D): the probability of hypothesis h given the data D. This is known as posterior probability.
* P(D|h): the probability of data d given that the hypothesis h was true. This is known as posterior probability.

It works in three steps:

1. Convert the data set into a frequency table.
2. Create Likelihood table by finding the probabilities.
3. Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

**Pros and Cons :**

Pros :

* It is easy and fast to predict class of test data set. It also perform well in multi class prediction
* When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
* It perform well in case of categorical input variables compared to numerical variable

Cons :

* If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”.
* Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

**Types of Naive Bayes Models :**

1. [Gaussian:](http://scikit-learn.org/stable/modules/naive_bayes.html) It is used in classification and it assumes that features follow a normal distribution.
2. [Multinomial](http://scikit-learn.org/stable/modules/naive_bayes.html)**:** It is used for discrete counts.
3. [Bernoulli](http://scikit-learn.org/stable/modules/naive_bayes.html)**:**The binomial model is useful if your feature vectors are binary (i.e. zeros and ones).

**2.7 Algorithm 2 – Random Forest Classifier**

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of majority voting method.

Random forest algorithm is an **ensemble classification** algorithm. Ensemble classifier means a group of classifiers. Instead of using only one classifier to predict the target, In ensemble, we use multiple classifiers to predict the target.

It works in four steps:

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.
3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.

**Pros and Cons :**

Pros :

* Random forests is considered as a highly accurate and robust method because of the number of decision trees participating in the process.
* It does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.
* The algorithm can be used in both classification and regression problems.
* Random forests can also handle missing values.

Cons :

* Random forests is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming.
* The model is difficult to interpret compared to a decision tree, where you can easily make a decision by following the path in the tree.

**2.8 Comparison between Naive Bayes and Random Forest Classifier**

|  |  |  |
| --- | --- | --- |
| **Sl .No** | **Naive Bayes** | **Random Forest** |
| 01 | It is a classification technique based on Bayes Theorem. | It is an ensemble classifier based on decision trees. |
| 02 | Can be used only for Classification. | Can be used for both Classification and Regression. |
| 03 | Naive Bayes model size is low and quite constant with respect to the data. | Random Forest model size is very large |
| 04 | When the data is dynamic and keeps changing, Naive Bayes can adapt quickly to the changes and new data. | For dynamic data, Random Forest you would have to rebuild the forest every time something changes. |

**2.9 Model accuracy and precision**

Before understanding the concept of how to analyse a model, we must know certain terms that help to determine the accuracy and precision of a model. These terms are as follows :

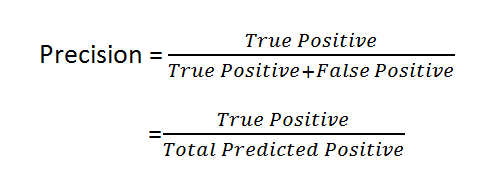
* True positive - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes.
* True negative - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no.
* False positive - When actual class is no and predicted class is yes.
* False negative - When actual class is yes but predicted class in no.

**The parameters that evaluate the performance of a model are :**

1. **Accuracy**
2. **Precision**
3. **Recall**
4. **F1 score**

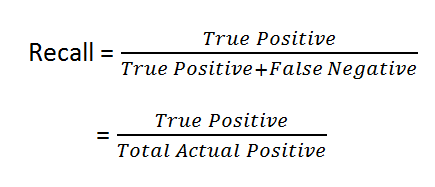
**Lets understand what each of these terms mean –**

1. **Accuracy** - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. We may think that if we have high accuracy then our model is best, but only when we have symmetric datasets where values of false positive and false negatives are almost same.
2. **Precision** - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Precision refers to how precise the model is out of those predicted positive, how many of them are actually positive.



1. **Recall**- Recall is the ratio of correctly predicted positive observations to the all observations in actual class. Recall is also known as **sensitivity**or **true positive rate**.

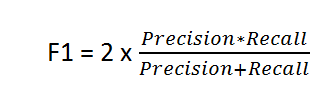
Recall actually calculates how many of the Actual Positives our model capture through labelling it as Positive (True Positive).



Precision and Recall can sound but confusing, but we must understand that - precision is a measure of how good predictions are with regard to false positives, whereas recall is measures how good the predictions are with regard to false negatives.

1. **F1 score** - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution.

Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall



F1 Score is needed when you want to seek a balance between Precision and Recall.

Together these terms give the grade score of how well the models performs for the given set of training data. Greater the score , better the efficiency of the model.

**2.10 Python GUI**

**GUI** is a desktop app which helps you to interact with the computers. They are used to perform different tasks in the desktops, laptops, other electronic devices, etc.

Python provides various interfaces to develop graphical user interfaces(GUIs).

* Tkinter - it is the Python interface to the Tk GUI toolkit shipped with Python.
* wxPython - This is an open-source Python interface for wxWindows
* JPython - JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine

Among these Tkinter is the most widely used GUI development interface.

**Tkinter :**

* **Tkinter** is an inbuilt **Python** module used to create simple **GUI** apps.
* It is the most commonly used module for **GUI** apps in the **Python**. It is the standard GUI library for Python.
* Python when combined with Tkinter provides a fast and easy way to create GUI applications.
* Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

1. Import the *Tkinter* module.
2. Create the GUI application main window.
3. Add one or more of the above-mentioned widgets to the GUI application.
4. Enter the main event loop to take action against each event triggered by the user.

Tkinter Widgets :

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter.

1. **Button** - used to display buttons in your application.
2. **Canvas** - used to draw shapes, such as lines, ovals, polygons and rectangles
3. **Checkbutton** - used to display a number of options as checkboxes
4. **Entry** - used to display a single-line text field for accepting values from user.
5. **Frame** - used as a container widget to organize other widgets.
6. **Label** - used to provide a single-line caption for other widgets.
7. **Listbox** - used to provide a list of options to a user.
8. **Menubutton** - used to display menus in your application.
9. **Menu** - used to provide various commands to a user.
10. **Message** - used to display multiline text fields for accepting values from user.
11. **Radiobutton** - used to display a number of options as radio buttons.
12. **Scale** - used to provide a slider widget.
13. **Scrollbar** - used to add scrolling capability to various widgets
14. **Text** - used to display text in multiple lines.
15. **Toplevel** - used to provide a separate window container.

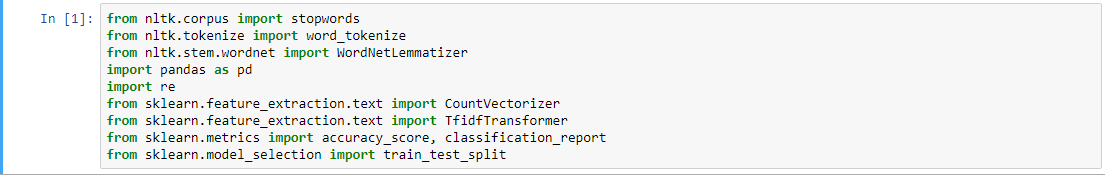
**CODING AND OUTPUT**

**3.1 Training and testing**

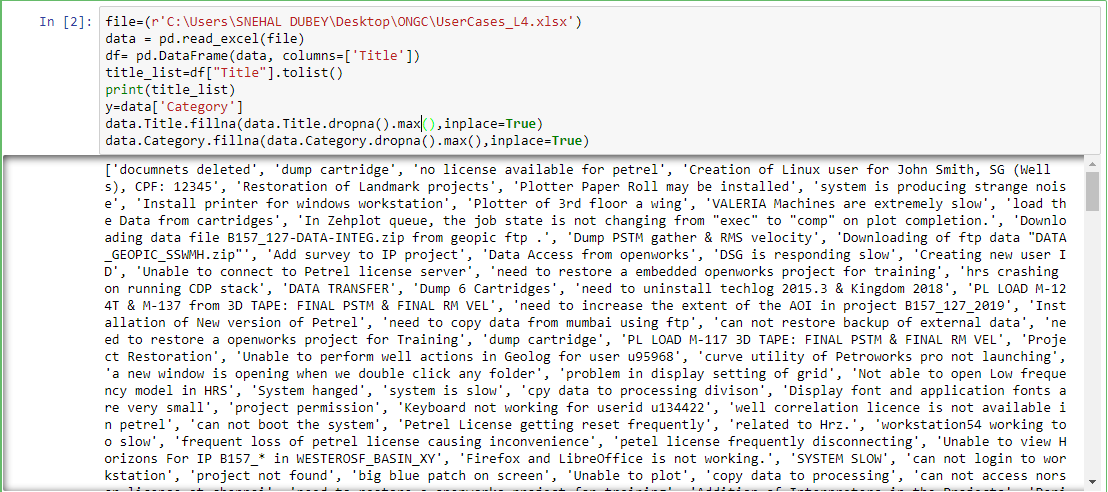
* Python version : 3.6
* Interface used : Jupyter Notebook
* Total columns : 4 (Sl.No, Title, Description, Category)
* Total rows : 500

Steps followed -

**1. Import the required libraries.**



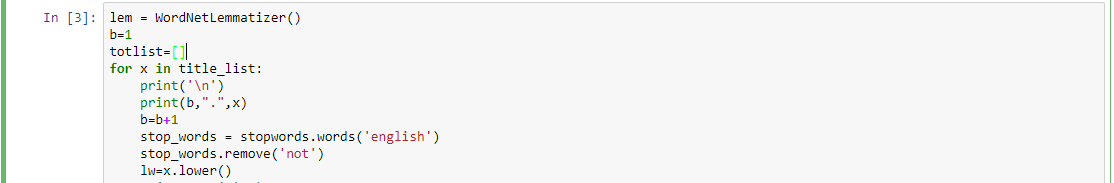
**2.Read the file.**



**3.Apply the steps of data cleaning – Our input was the ‘Title’ provided in the file.**

* **Remove the stopwords and convert to lowercase.**

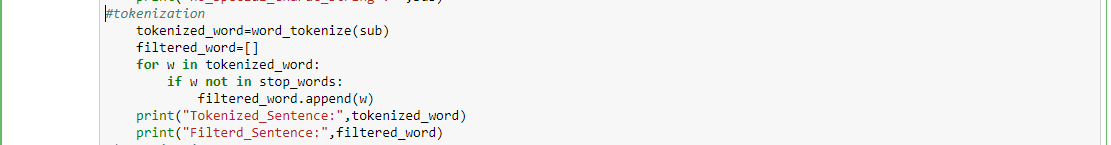
(‘not’ is also a stopword and when removed it changed the entire meaning of the content. So we had to include it in the code. )



* **Removing the special characters.**

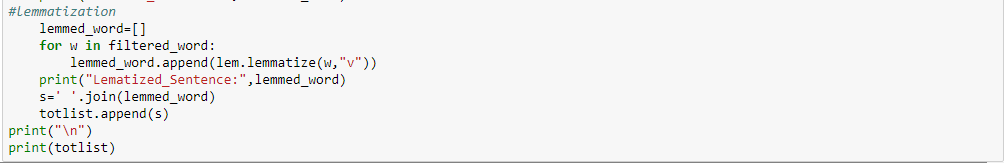


* **Tokenize to extract the keywords.**

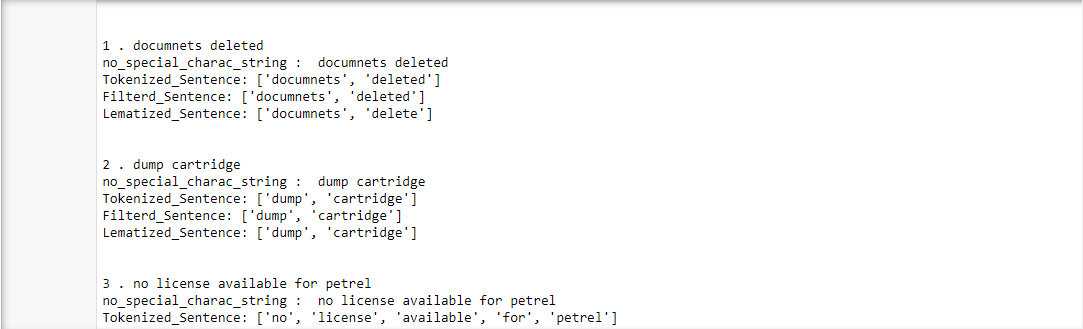


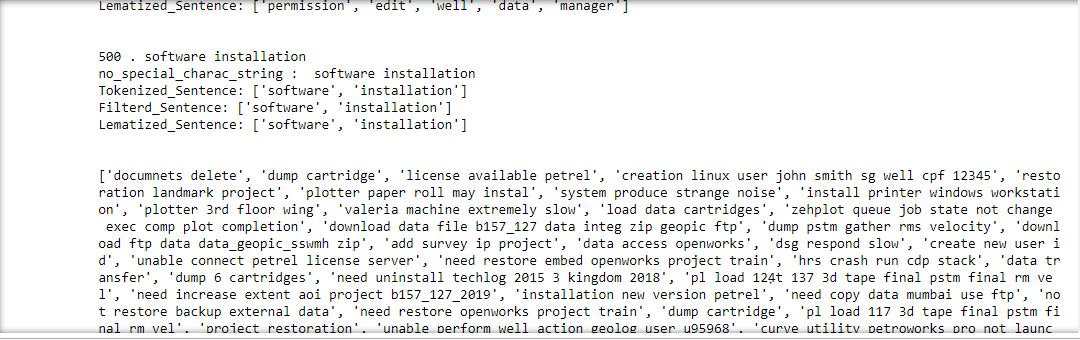
* **Lemmatize the tokenized words.**

(If we applied ‘stemming’, words like ‘issue’ , ‘license’, etc became ‘issu’, ‘licens’ respectively. Therefore , we only lemmatized the words.)

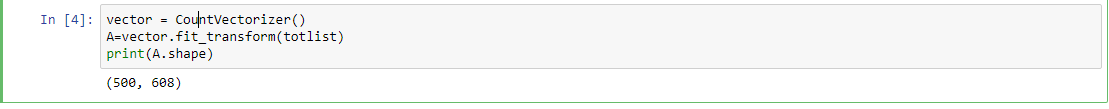


**Output (step 3):**





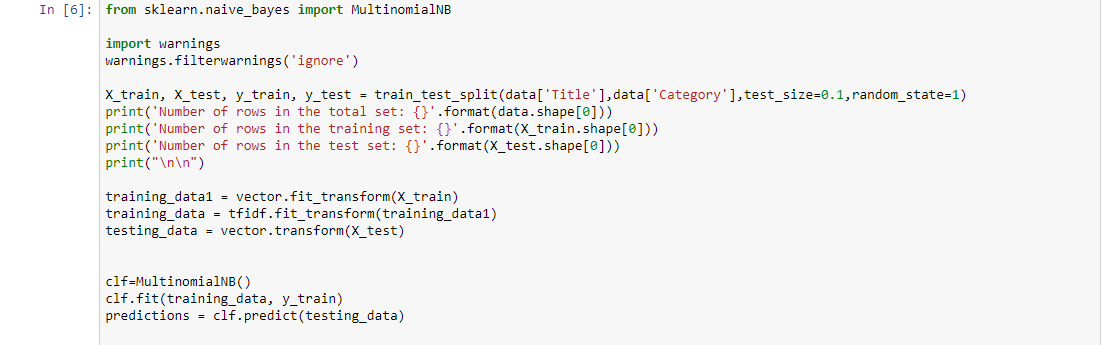
**4.Vectorize the words**

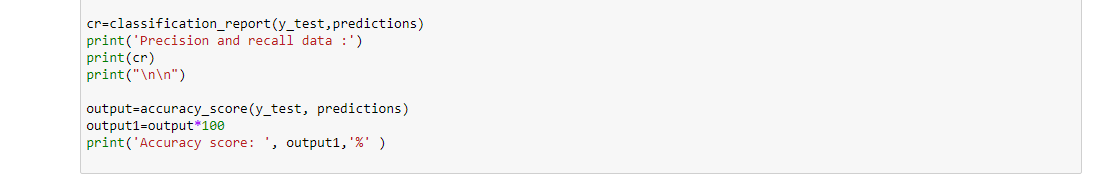


**5.Finding the Term Frequency(TF) and Inverse Document Frequency(IDF)**

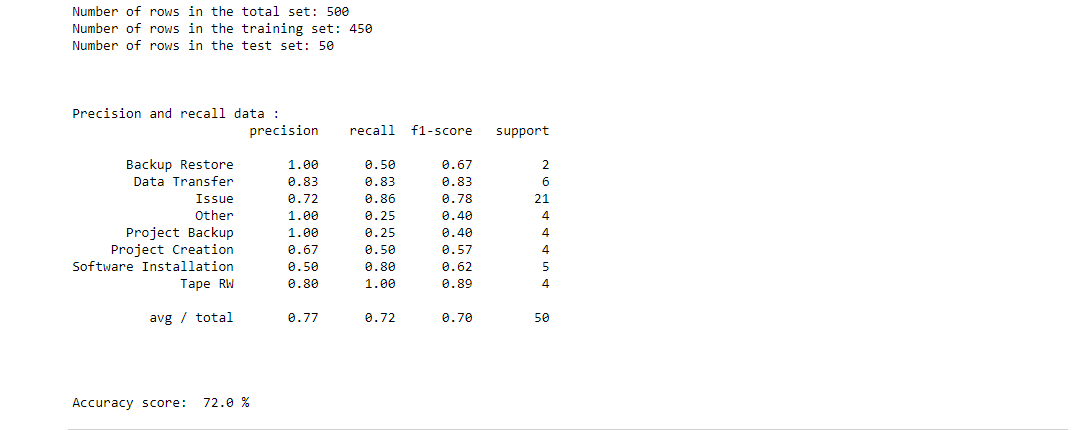


**6.Naive Bayes Classifier –**

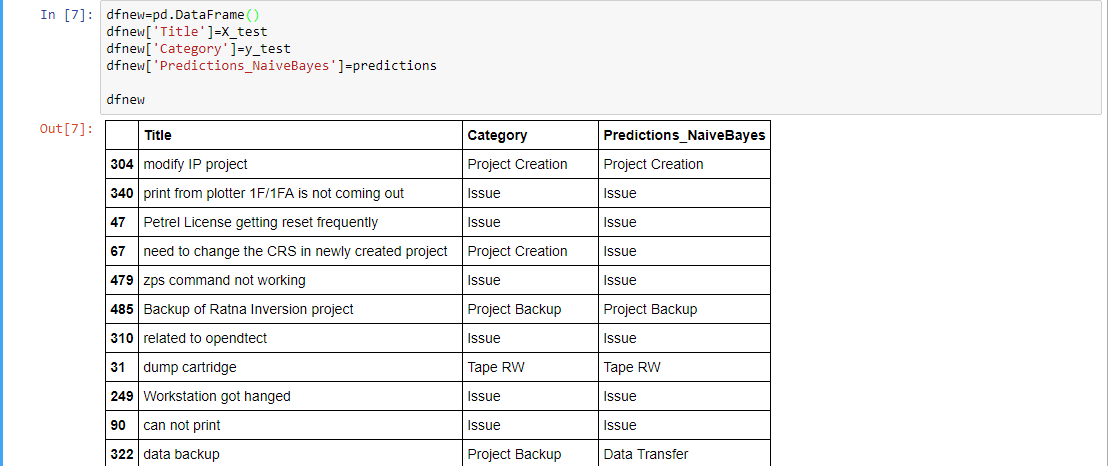
* **Train\_split and Test\_spl**
* **Measure accuracy and precision**



**Output(step 6) :**

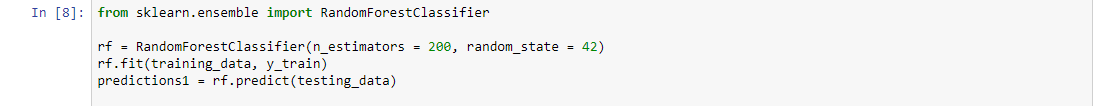


**7.Viewing the result(NaiveBayes)**

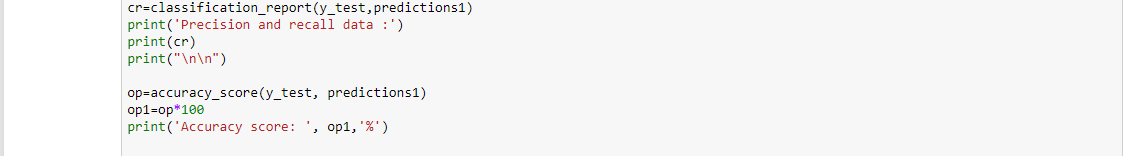


**8.Random Forest Classifier**

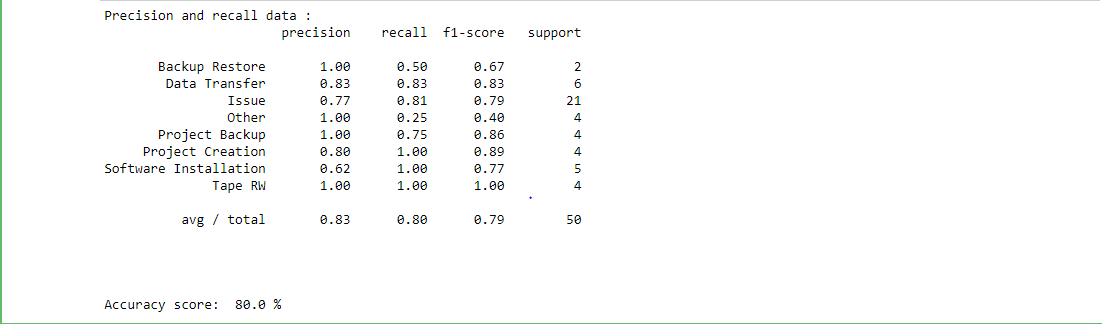
* **Train\_split and test\_split**



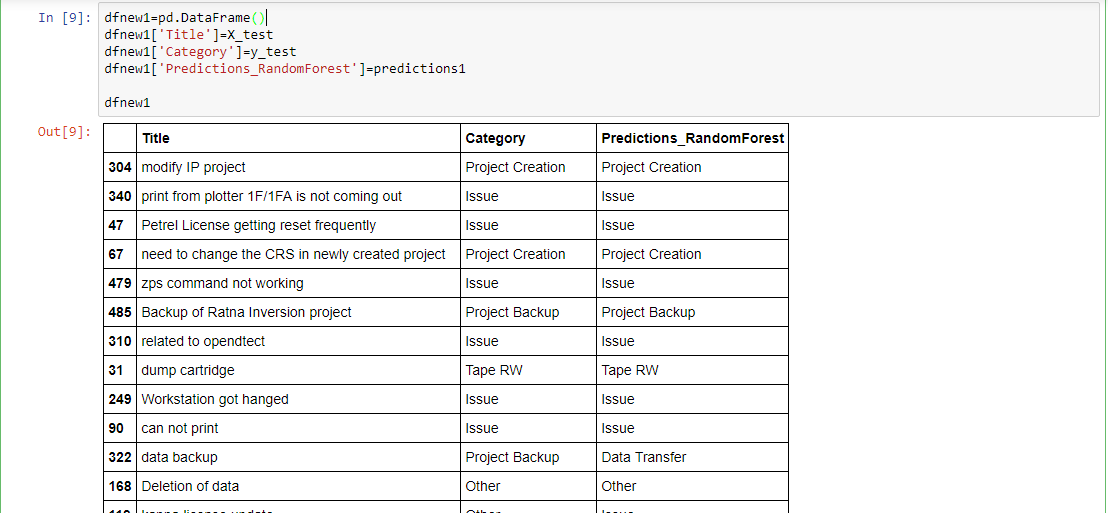
* **Model accuracy and precision**



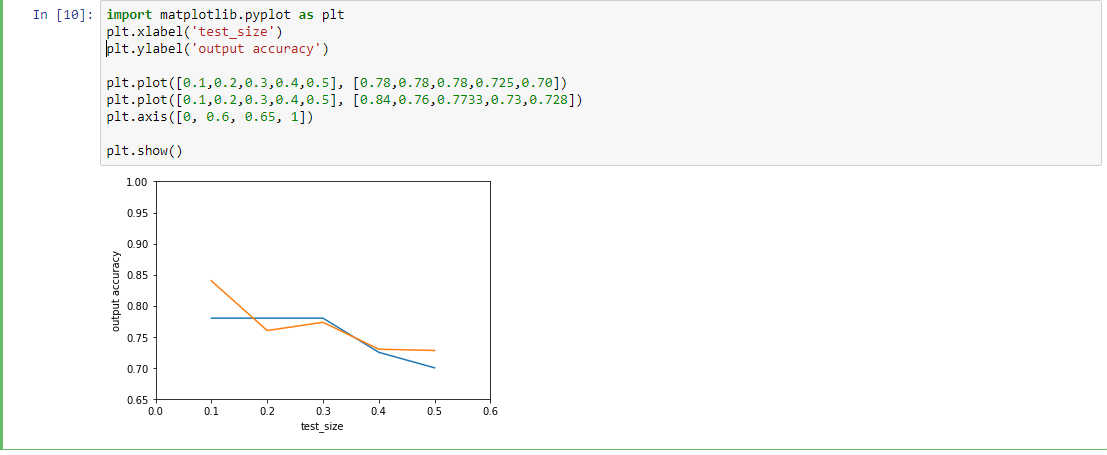
**Output (step 8):**



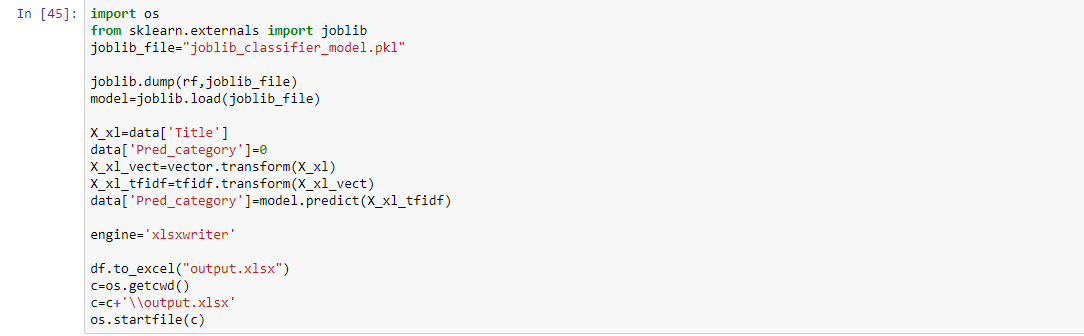
**9.Viewing the result(RandomForest).**

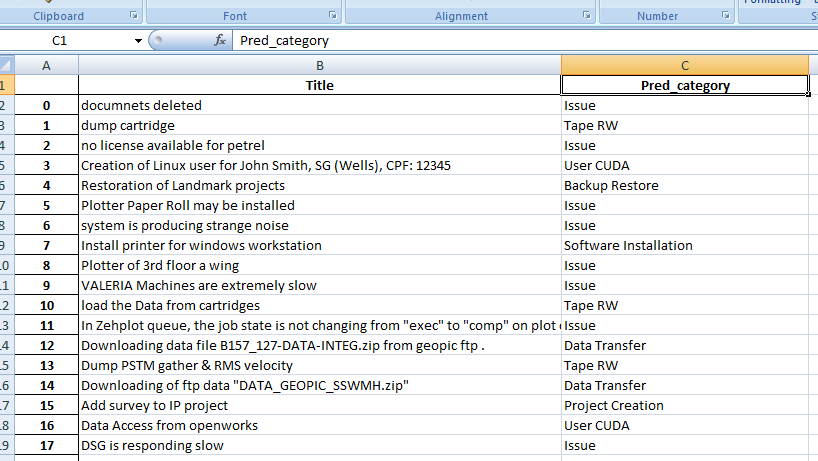


**10.Plotting the accuracy graph**



**11.Saving and loading the file for future use**

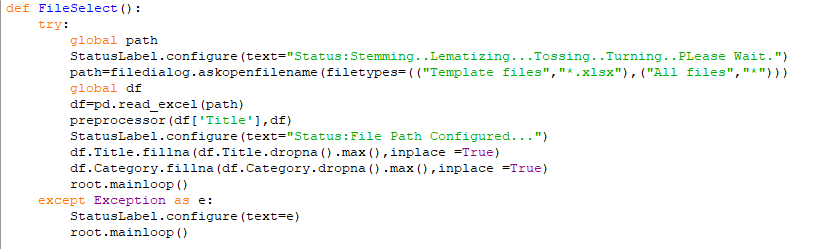




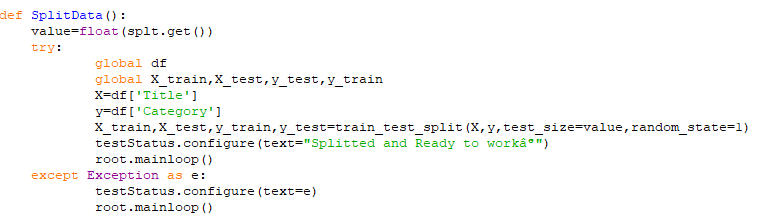
-----------------------

**3.2 GUI**

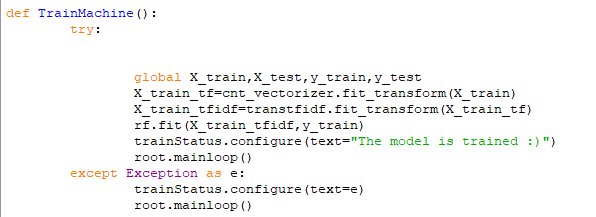
**1.Select file**

****

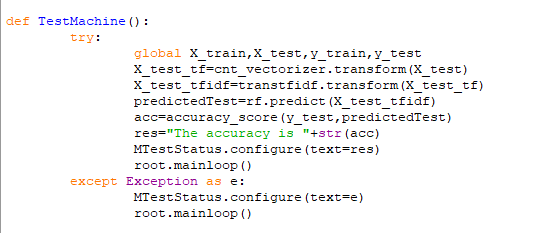
**2.Spliting the input data**

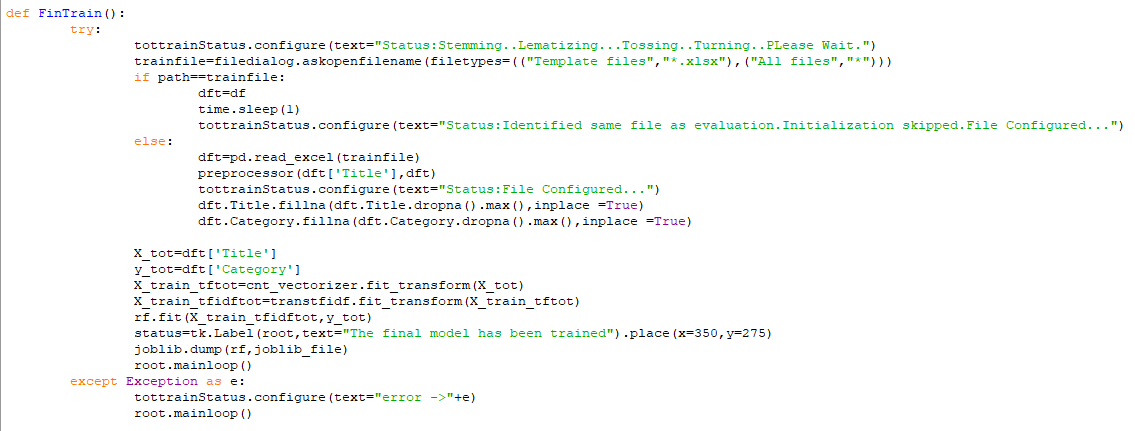
****

**3.Train the model**

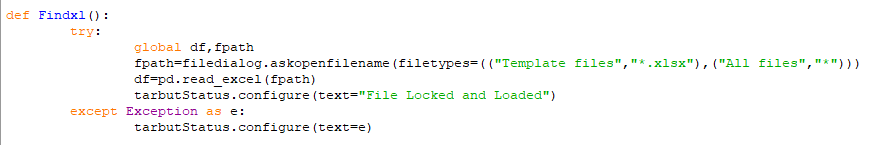
****

**4. Test the model**

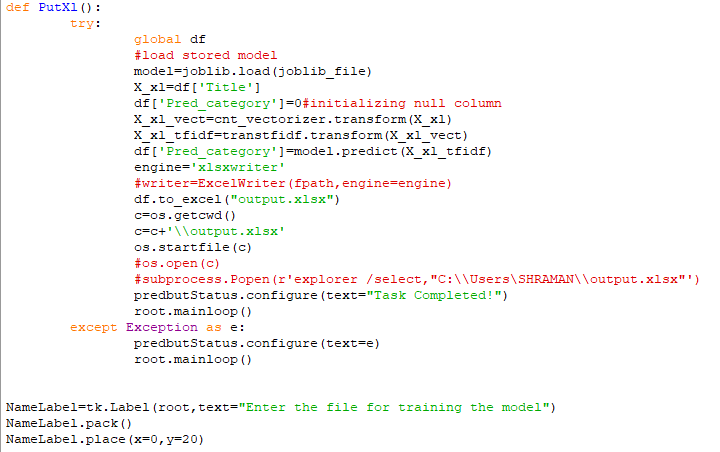
****

**5. Final Training of machine**

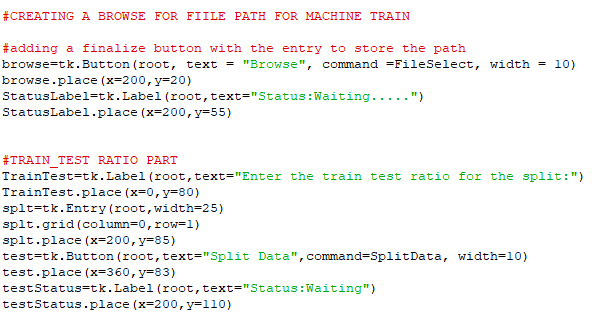
**6. Defining path**

****

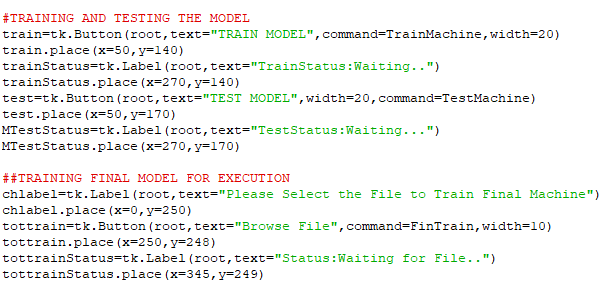
**7.Storing the predicted category in Excel file**

****

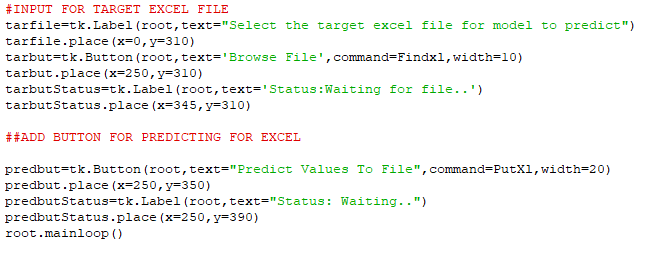
**8.Add ‘Browse’ button to interface**

****

**9.Add ‘Train’ and ‘Test’ buttons on interface**

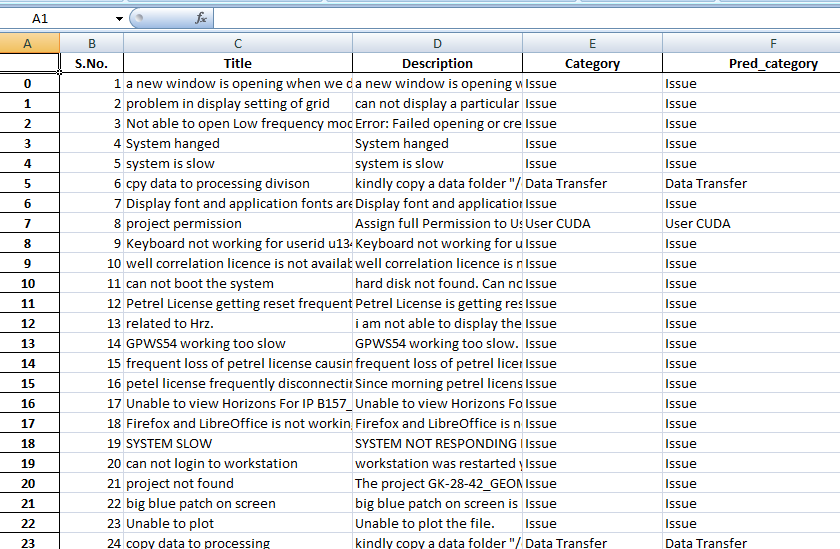
****

**10.Add ‘Input ’ button to select target file and ‘Predict’ button to get the result**

****

**Output of GUI –**

****

****

**CONCLUSION**

Machine learning is the ultimate tool to get the best out of a machine. Applying the various machine learning algorithms we can design a machine to work as per our requirement.

To make the machine able to predict the correct output, two important algorithms have been implemented here-

* Naive Bayes
* Random Forest

Out of 500 test cases, we had split up the test data in varying proportions, from 10% to 50% of the total data. Each time we got a different value of accuracy. The more we train the machine, the better is its capability to predict accurately. When we trained the machine with 90% of total data (or tested it with 10% of data), we achieved the maximum accuracy.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl. No** | **Test size** | **Rows in Training set** | **Rows in Testing set** | **Naive Bayes accuracy** | **Random Forest accuracy** |
| **1** | 0.1 | 450 | 50 | 78% | 84 % |
| **2** | 0.2 | 400 | 100 | 78 % | 76 % |
| **3** | 0.3 | 350 | 150 | 78 % | 77.33 % |
| **4** | 0.4 | 300 | 200 | 72.5 % | 73 % |
| **5** | 0.5 | 250 | 250 | 70 % | 72.8 % |

The GUI interface will help the user to easily use the machine for any future use, without having to look at the code. It will train the machine on any excel file we choose , will retain the result and finally it will correctly predict the unlabelled dataset. When tested on various datasets, the machine was able to produce satisfactory results every time.

We have tried to use other algorithms too like XGBoost, SVM Classifier. But those algorithms have very low accuracies.

**FUTURE SCOPE**

This project aims at classifying issues raised by employees into their specific categories. Currently the accuracy achieved is in the range of 70-80 %. In future we aim to increase it to 80-90%. It can be achieved in the following ways:

* In this project, the algorithms applied take the words individually and assign weights to them accordingly. Words will be taken in association with their affinity to each other and then assigning weights to them can increase the efficiency to a step further.
* Text classification and analysis can be implemented using various other algorithms of Neural Networks such as Glove.
* In this project classification was done on single sentences and not over a paragraph, machine can be modified to classify a text document given in the form of a paragraph using the concept of ‘Sentence Tokenization’.
* We can also make the machine to work on a variety of areas. Generic model can be made to further work on different sectors as required.
* Transfer learning can also be implemented if a suitable machine model trained is found.
* We can also use N-Grams, i.e., a set of N successive words to prevent the meaning change when considering only single words in a text document.
* However, we can always improve the accuracy of the model by tuning the hyperparameters of the model or the count vectorizer or the tf-idf model.  
  We can also tune the parameters of the Random Forest Classifier to improve the overall accuracy

However, all of this accuracy improving techniques boils down to a simple “No Free Lunch” Theorem which states that “If an algorithm performs better than random search on some class of problems then in must perform worse than random search on the remaining problems.”

**REFRENCES**

1. Text Analytics for Beginners using NLTK, DataCamp, Avinash Navlani September 4th, 2018.[Online].Available:<https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk>
2. Text Classification with Python and Scikit-Learn, StackAbuse , [Usman Malik](https://twitter.com/usman_malikk) , August 27, 2018. [Online]. Available :<https://stackabuse.com/text-classification-with-python-and-scikit-learn/>
3. Accuracy, Precision, Recall or F1?, TowardsDataScience, [Koo Ping Shung](https://towardsdatascience.com/@koolanalytics), Mar 15, 2018.[Online]. Available : <https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9>
4. Performance measures in Azure ML: Accuracy, Precision, Recall and F1 Score, Microsoft Developer, [Andreas de Ruiter (Microsoft)](https://social.msdn.microsoft.com/profile/Andreas+de+Ruiter+%28Microsoft%29), February 9, 2015. [Online]. Available:<https://blogs.msdn.microsoft.com/andreasderuiter/2015/02/09/performance-measures-in-azure-ml-accuracy-precision-recall-and-f1-score/>

# Naive Bayes Classification using Scikit-learn, DataCamp, Avinash Navlani September4th,2018.[Online].Available: <https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn>

# 