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)** and Project 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 and text analytics 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**

INTRODUCTION

1.1 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.

1.2 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, emotion and 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.

1.3 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.
2) **Converting into lower case :** entire text in converted into lower case for uniformity.
3) **Removing punctuation marks :** since punctuations and symbols are not providing any useful information , they are removed.

4) **Removing Stopwords :** Stopwords are the words that frequently occur in English, and do not have any semantics ; like – ‘is’, ‘are’, ‘were’, ‘was’, ‘the’ etc.
5) **Stemming :** It refers to reduction of words to their root by dropping unnecessary characters , especially suffixes(‘-s’, ‘-es’).
6) **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 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.

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- 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 :

a. Gaussian: It is used in classification and it assumes that features follow a normal distribution.

b. Multinomial**:** It is used for discrete counts.

c. Bernoulli**:** 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.

$$\boxed{\begin{aligned}\text{Precision} &= \frac{True\ Positive}{True\ Positive + False\ Positive}\\ &= \frac{True\ Positive}{Total\ Predicted\ Positive}\end{aligned}}$$

3. **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).

$$\boxed{\begin{aligned}\text{Recall} &= \frac{True\ Positive}{True\ Positive + False\ Negative}\\ &= \frac{True\ Positive}{Total\ Actual\ Positive}\end{aligned}}$$

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.

4. **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

$$\boxed{F1 = 2 \times \frac{Precision*Recall}{Precision+Recall}}$$

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 −

i. Import the *Tkinter* module.

ii. Create the GUI application main window.

iii. Add one or more of the above-mentioned widgets to the GUI application.

iv. 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.

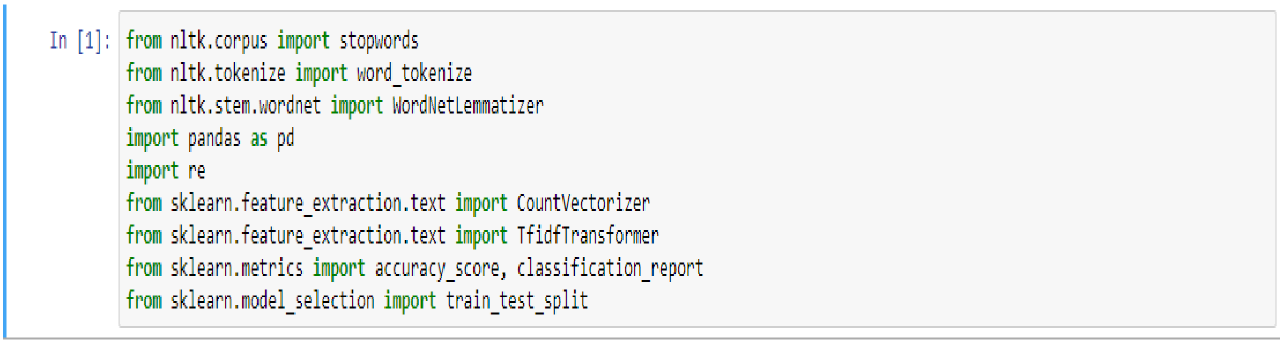
CODING AND OUTPUT

3.1 Training and testing

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

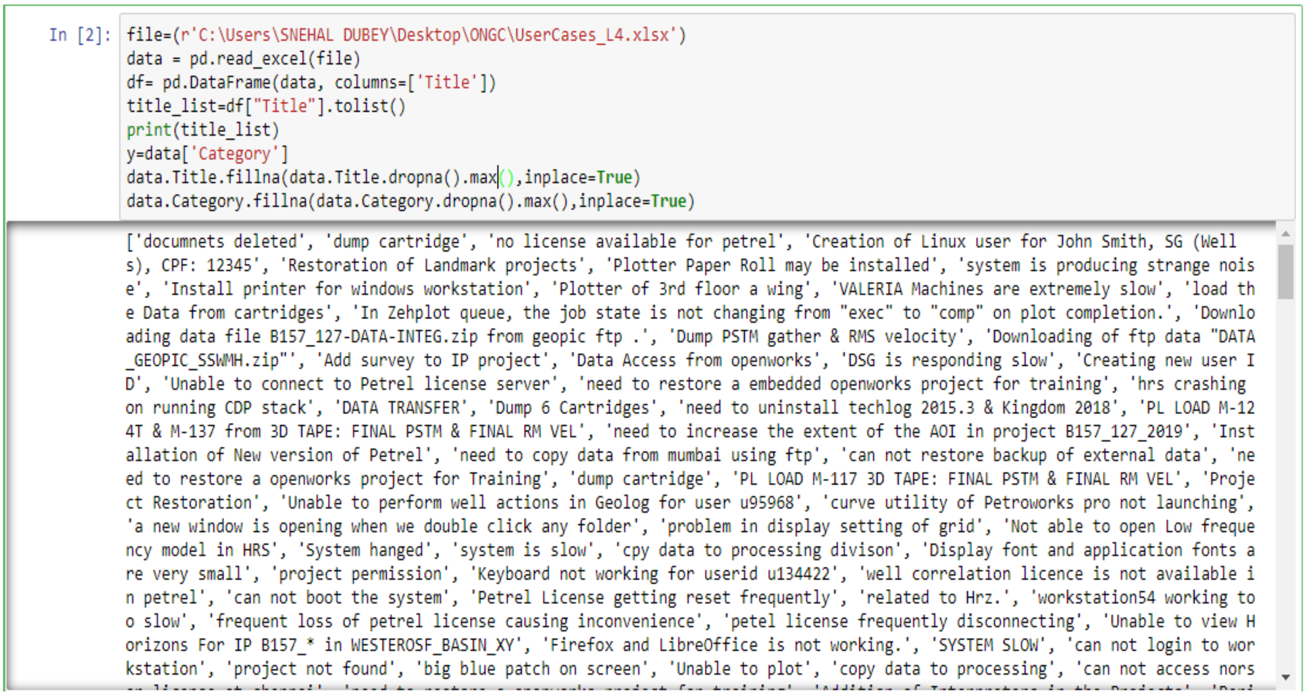
Steps followed -

1. Import the required libraries.



```
In [1]: from nltk.corpus import stopwords
        from nltk.tokenize import word_tokenize
        from nltk.stem.wordnet import WordNetLemmatizer
        import pandas as pd
        import re
        from sklearn.feature_extraction.text import CountVectorizer
        from sklearn.feature_extraction.text import TfidfTransformer
        from sklearn.metrics import accuracy_score, classification_report
        from sklearn.model_selection import train_test_split
```

2.Read the file.



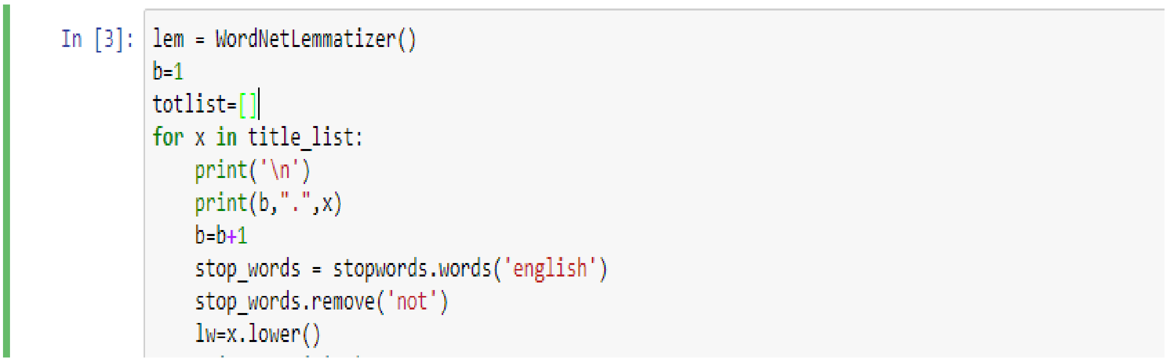
```
In [2]: file=(r'C:\Users\SNEHAL DUBEY\Desktop\ONGC\UserCases_L4.xlsx')
        data = pd.read_excel(file)
        df= pd.DataFrame(data, columns=['Title'])
        title_list=df["Title"].tolist()
        print(title_list)
        y=data['Category']
        data.Title.fillna(data.Title.dropna().max(),inplace=True)
        data.Category.fillna(data.Category.dropna().max(),inplace=True)
```

```
['documnets deleted', 'dump cartridge', 'no license available for petrel', 'Creation of Linux user for John Smith, SG (Well
s), CPF: 12345', 'Restoration of Landmark projects', 'Plotter Paper Roll may be installed', 'system is producing strange nois
e', 'Install printer for windows workstation', 'Plotter of 3rd floor a wing', 'VALERIA Machines are extremely slow', 'load th
e Data from cartridges', 'In Zehplot queue, the job state is not changing from "exec" to "comp" on plot completion.', 'Downlo
ading data file B157_127-DATA-INTEG.zip from geopic ftp .', 'Dump PSTM gather & RMS velocity', 'Downloading of ftp data "DATA
_GEOPIC_SSWMH.zip"', 'Add survey to IP project', 'Data Access from openworks', 'DSG is responding slow', 'Creating new user I
D', 'Unable to connect to Petrel license server', 'need to restore a embedded openworks project for training', 'hrs crashing
on running CDP stack', 'DATA TRANSFER', 'Dump 6 Cartridges', 'need to uninstall techlog 2015.3 & Kingdom 2018', 'PL LOAD M-12
4T & M-137 from 3D TAPE: FINAL PSTM & FINAL RM VEL', 'need to increase the extent of the AOI in project B157_127_2019', 'Inst
allation of New version of Petrel', 'need to copy data from mumbai using ftp', 'can not restore backup of external data', 'ne
ed to restore a openworks project for Training', 'dump cartridge', 'PL LOAD M-117 3D TAPE: FINAL PSTM & FINAL RM VEL', 'Proje
ct Restoration', 'Unable to perform well actions in Geolog for user u95968', 'curve utility of Petroworks pro not launching',
'a new window is opening when we double click any folder', 'problem in display setting of grid', 'Not able to open Low freque
ncy model in HRS', 'System hanged', 'system is slow', 'cpy data to processing divison', 'Display font and application fonts a
re very small', 'project permission', 'Keyboard not working for userid u134422', 'well correlation licence is not available i
n petrel', 'can not boot the system', 'Petrel License getting reset frequently', 'related to Hrz.', 'workstation54 working to
o slow', 'frequent loss of petrel license causing inconvenience', 'petel license frequently disconnecting', 'Unable to view H
orizons For IP B157_* in WESTEROSF_BASIN_XY', 'Firefox and LibreOffice is not working.', 'SYSTEM SLOW', 'can not login to wor
kstation', 'project not found', 'big blue patch on screen', 'Unable to plot', 'copy data to processing', 'can not access nors
```

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

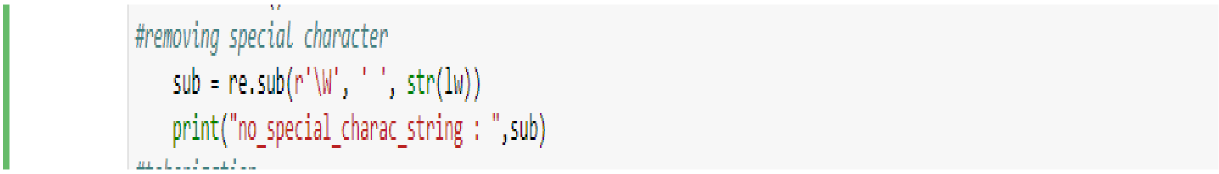
- **Remove the stopwords and convert to lowercase.**

(‘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.)



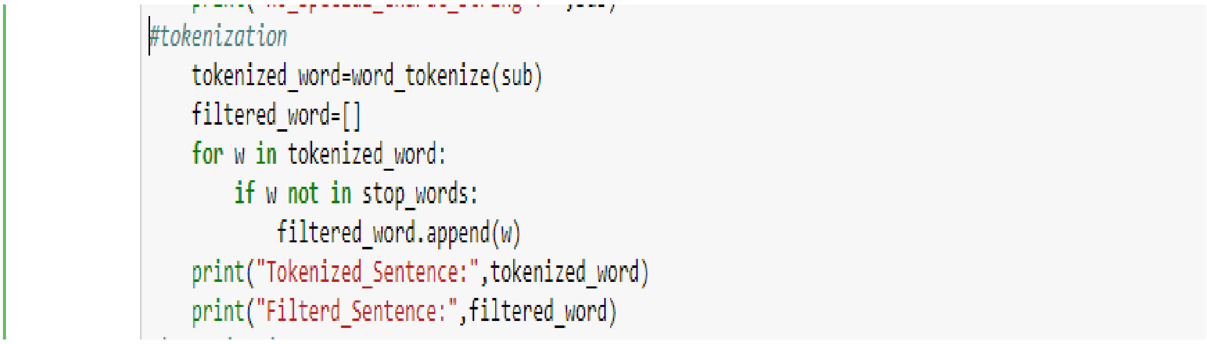
```
In [3]: lem = WordNetLemmatizer()
        b=1
        totlist=[]
        for x in title_list:
            print('\n')
            print(b,".",x)
            b=b+1
            stop_words = stopwords.words('english')
            stop_words.remove('not')
            lw=x.lower()
```

- **Removing the special characters.**



```
#removing special character
    sub = re.sub(r'\W', ' ', str(lw))
    print("no_special_charac_string : ",sub)
```

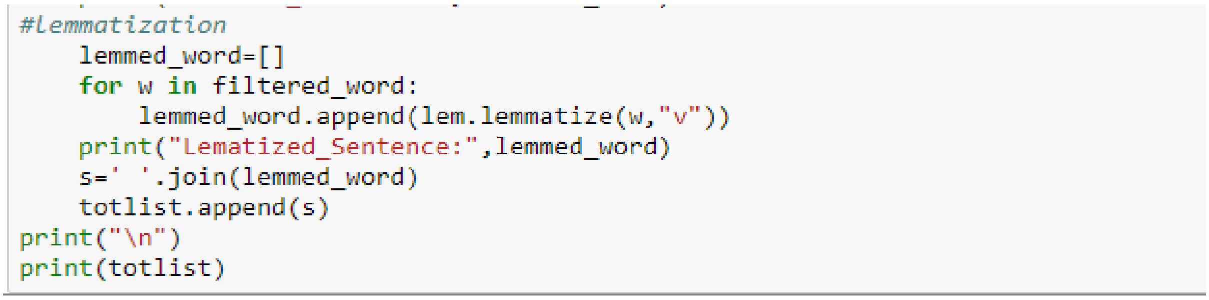
- **Tokenize to extract the keywords.**



```
#tokenization
    tokenized_word=word_tokenize(sub)
    filtered_word=[]
    for w in tokenized_word:
        if w not in stop_words:
            filtered_word.append(w)
    print("Tokenized_Sentence:",tokenized_word)
    print("Filterd_Sentence:",filtered_word)
```

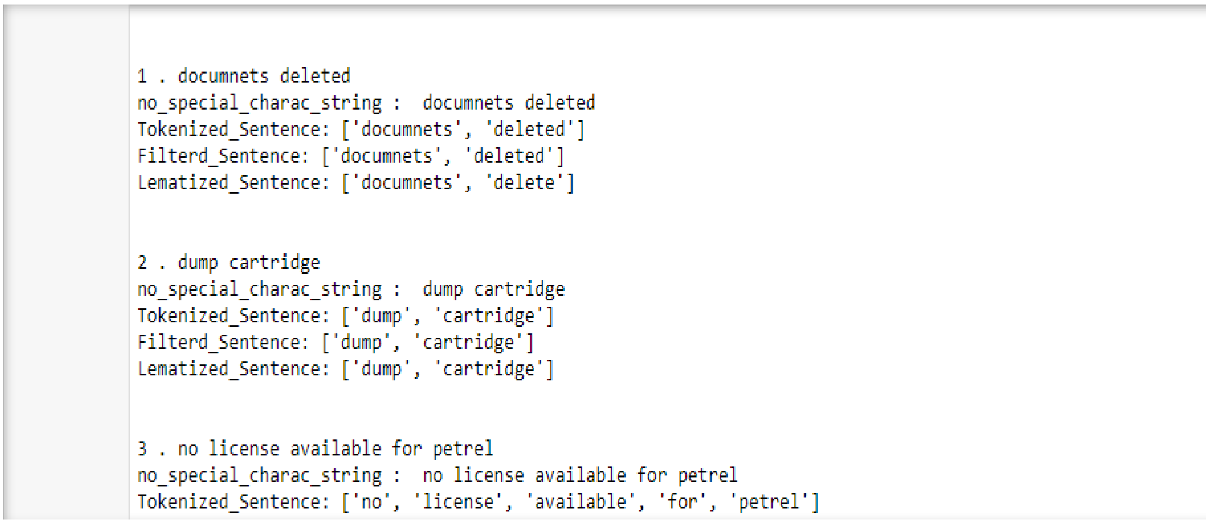
- **Lemmatize the tokenized words.**

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

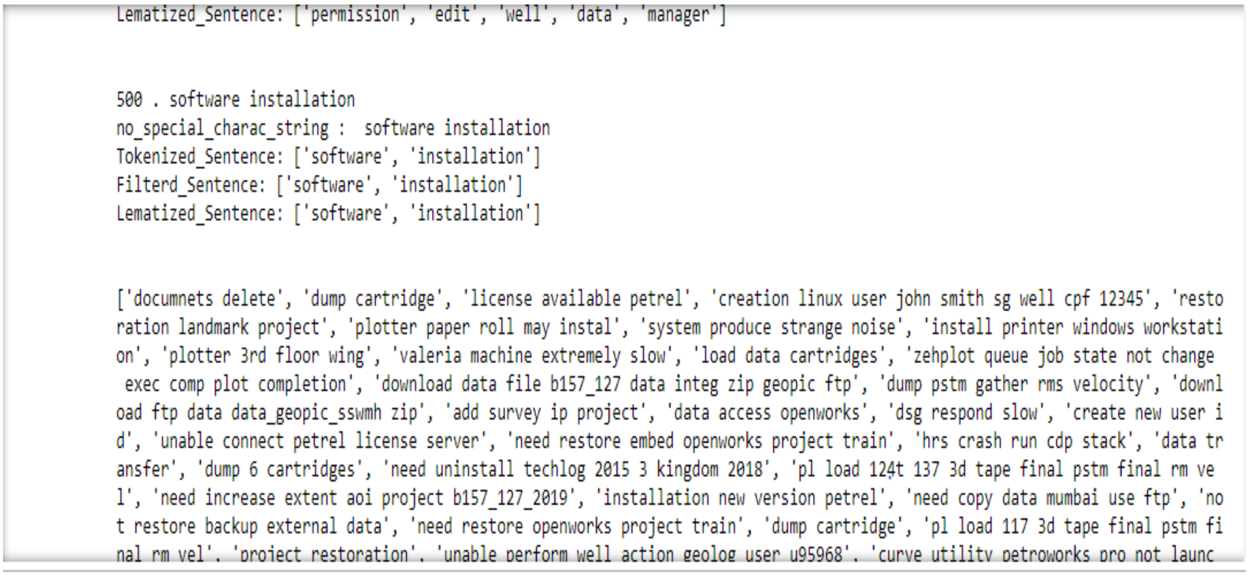


```
#Lemmatization
    lemmed_word=[]
    for w in filtered_word:
        lemmed_word.append(lem.lemmatize(w,"v"))
    print("Lematized_Sentence:",lemmed_word)
    s=' '.join(lemmed_word)
    totlist.append(s)
print("\n")
print(totlist)
```

Output (step 3):

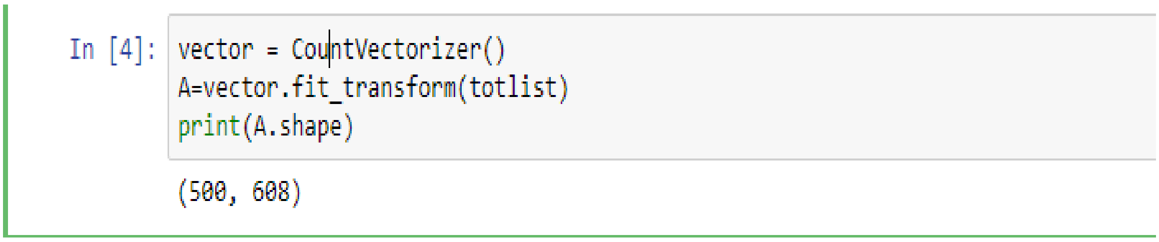


```
1 . documnets deleted
no_special_charac_string :  documnets deleted
Tokenized_Sentence: ['documnets', 'deleted']
Filterd_Sentence: ['documnets', 'deleted']
Lematized_Sentence: ['documnets', 'delete']


2 . dump cartridge
no_special_charac_string :  dump cartridge
Tokenized_Sentence: ['dump', 'cartridge']
Filterd_Sentence: ['dump', 'cartridge']
Lematized_Sentence: ['dump', 'cartridge']


3 . no license available for petrel
no_special_charac_string :  no license available for petrel
Tokenized_Sentence: ['no', 'license', 'available', 'for', 'petrel']
```



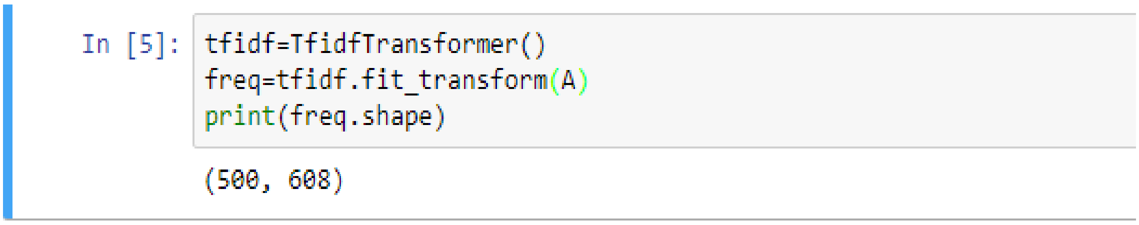
```
Lematized_Sentence: ['permission', 'edit', 'well', 'data', 'manager']


500 . software installation
no_special_charac_string :  software installation
Tokenized_Sentence: ['software', 'installation']
Filterd_Sentence: ['software', 'installation']
Lematized_Sentence: ['software', 'installation']


['documnets delete', 'dump cartridge', 'license available petrel', 'creation linux user john smith sg well cpf 12345', 'resto
ration landmark project', 'plotter paper roll may instal', 'system produce strange noise', 'install printer windows workstati
on', 'plotter 3rd floor wing', 'valeria machine extremely slow', 'load data cartridges', 'zehplot queue job state not change
 exec comp plot completion', 'download data file b157_127 data integ zip geopic ftp', 'dump pstm gather rms velocity', 'downl
oad ftp data data_geopic_sswmh zip', 'add survey ip project', 'data access openworks', 'dsg respond slow', 'create new user i
d', 'unable connect petrel license server', 'need restore embed openworks project train', 'hrs crash run cdp stack', 'data tr
ansfer', 'dump 6 cartridges', 'need uninstall techlog 2015 3 kingdom 2018', 'pl load 124t 137 3d tape final pstm final rm ve
l', 'need increase extent aoi project b157_127_2019', 'installation new version petrel', 'need copy data mumbai use ftp', 'no
t restore backup external data', 'need restore openworks project train', 'dump cartridge', 'pl load 117 3d tape final pstm fi
nal rm vel', 'project restoration', 'unable perform well action geolog user u95968', 'curve utility petroworks pro not launc
```

4.Vectorize the words



```
In [4]: vector = CountVectorizer()
        A=vector.fit_transform(totlist)
        print(A.shape)

        (500, 608)
```

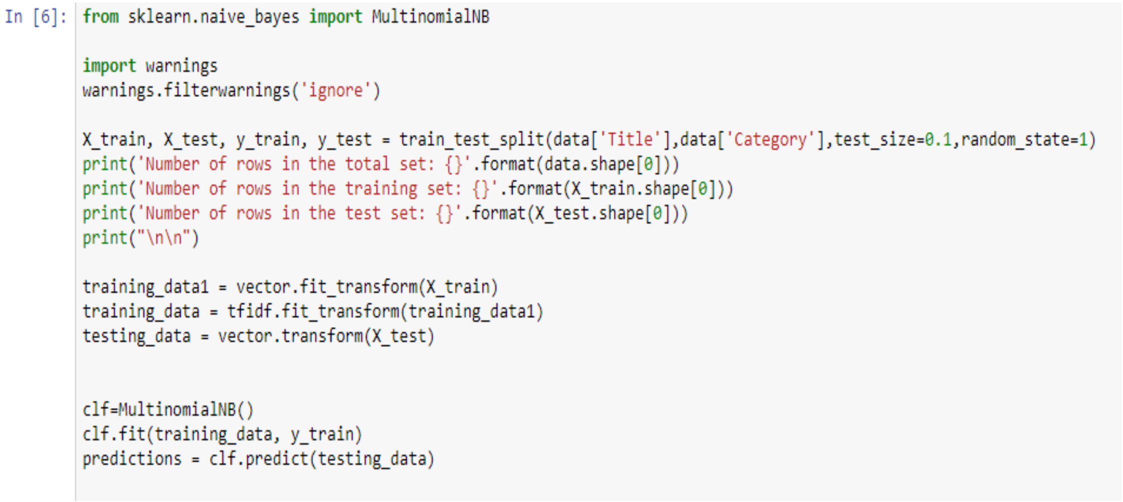
5.Finding the Term Frequency(TF) and Inverse Document Frequency(IDF)



```
In [5]: tfidf=TfidfTransformer()
        freq=tfidf.fit_transform(A)
        print(freq.shape)

        (500, 608)
```

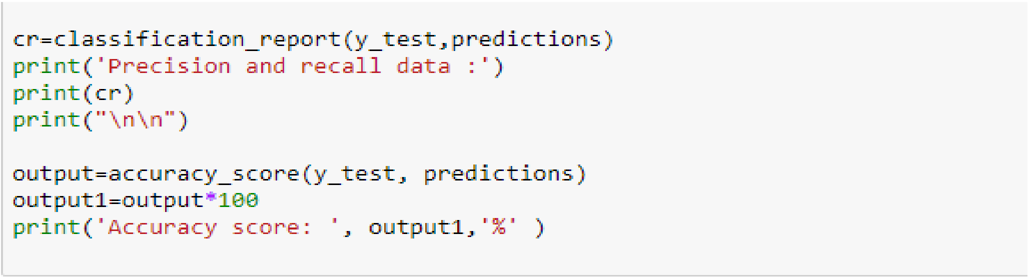
6.Naive Bayes Classifier –

- **Train_split and Test_spl**



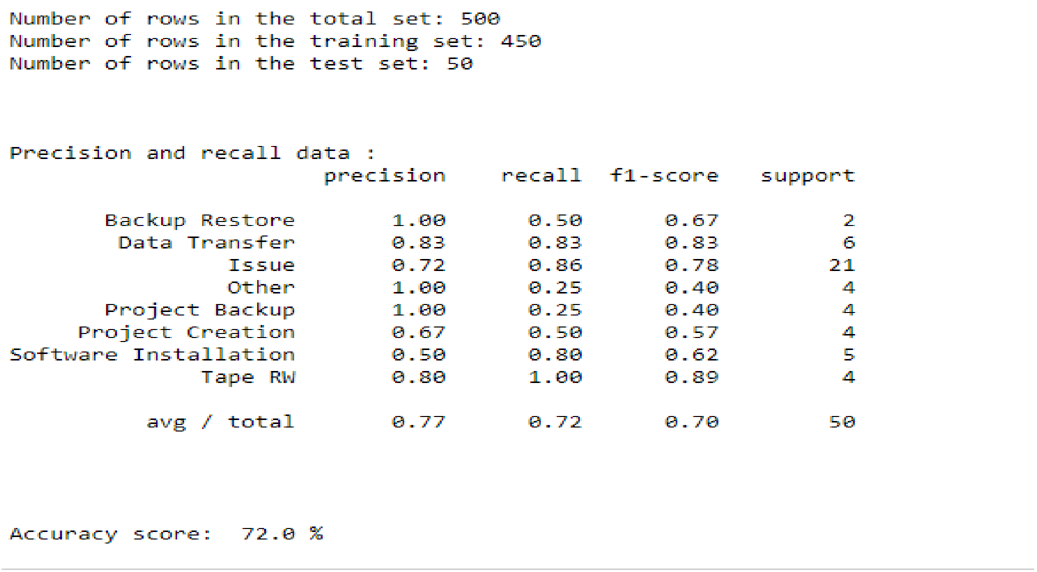
```
In [6]: from sklearn.naive_bayes import MultinomialNB

        import warnings
        warnings.filterwarnings('ignore')

        X_train, X_test, y_train, y_test = train_test_split(data['Title'],data['Category'],test_size=0.1,random_state=1)
        print('Number of rows in the total set: {}'.format(data.shape[0]))
        print('Number of rows in the training set: {}'.format(X_train.shape[0]))
        print('Number of rows in the test set: {}'.format(X_test.shape[0]))
        print("\n\n")

        training_data1 = vector.fit_transform(X_train)
        training_data = tfidf.fit_transform(training_data1)
        testing_data = vector.transform(X_test)


        clf=MultinomialNB()
        clf.fit(training_data, y_train)
        predictions = clf.predict(testing_data)
```

- **Measure accuracy and precision**



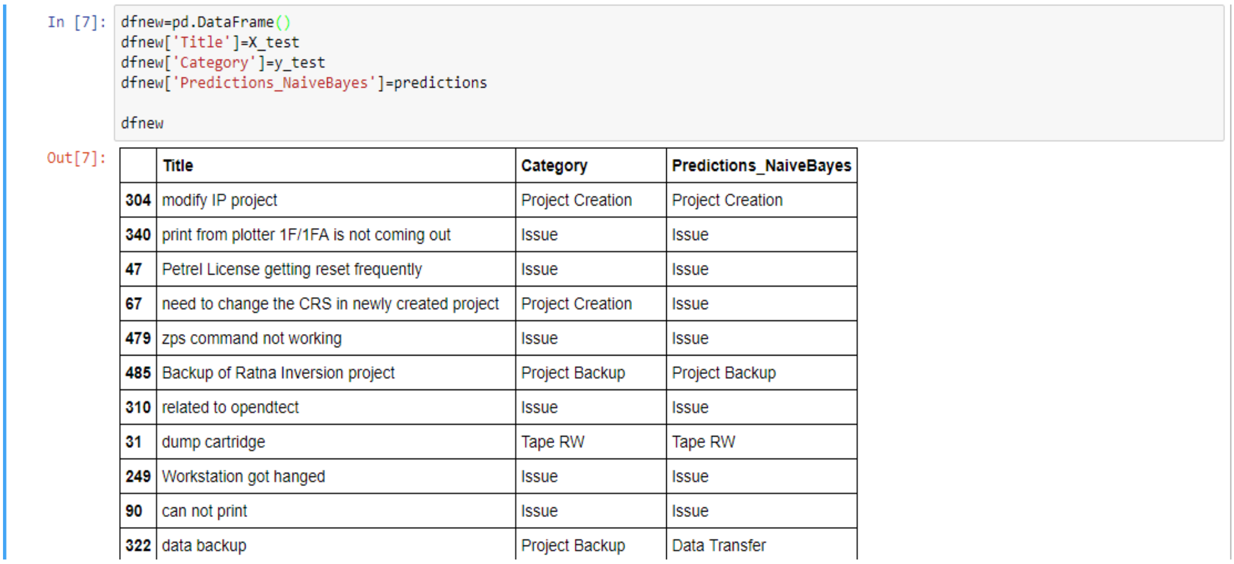
```
cr=classification_report(y_test,predictions)
print('Precision and recall data :')
print(cr)
print("\n\n")

output=accuracy_score(y_test, predictions)
output1=output*100
print('Accuracy score: ', output1,'%' )
```

Output(step 6) :



```
Number of rows in the total set: 500
Number of rows in the training set: 450
Number of rows in the test set: 50


Precision and recall data :
                      precision    recall  f1-score   support

       Backup Restore       1.00      0.50      0.67         2
        Data Transfer       0.83      0.83      0.83         6
                Issue       0.72      0.86      0.78        21
                Other       1.00      0.25      0.40         4
       Project Backup       1.00      0.25      0.40         4
     Project Creation       0.67      0.50      0.57         4
Software Installation       0.50      0.80      0.62         5
              Tape RW       0.80      1.00      0.89         4

          avg / total       0.77      0.72      0.70        50


Accuracy score:  72.0 %
```

7.Viewing the result(NaiveBayes)



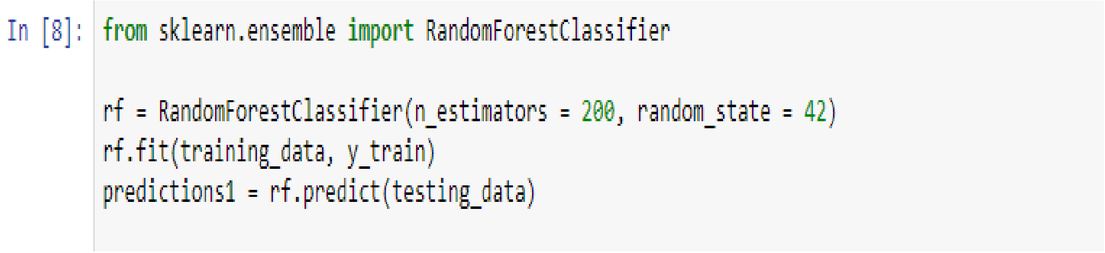
```
In [7]: dfnew=pd.DataFrame()
        dfnew['Title']=X_test
        dfnew['Category']=y_test
        dfnew['Predictions_NaiveBayes']=predictions

        dfnew
```

Out[7]:

	Title	Category	Predictions_NaiveBayes
304	modify IP project	Project Creation	Project Creation
340	print from plotter 1F/1FA is not coming out	Issue	Issue
47	Petrel License getting reset frequently	Issue	Issue
67	need to change the CRS in newly created project	Project Creation	Issue
479	zps command not working	Issue	Issue
485	Backup of Ratna Inversion project	Project Backup	Project Backup
310	related to opendtect	Issue	Issue
31	dump cartridge	Tape RW	Tape RW
249	Workstation got hanged	Issue	Issue
90	can not print	Issue	Issue
322	data backup	Project Backup	Data Transfer

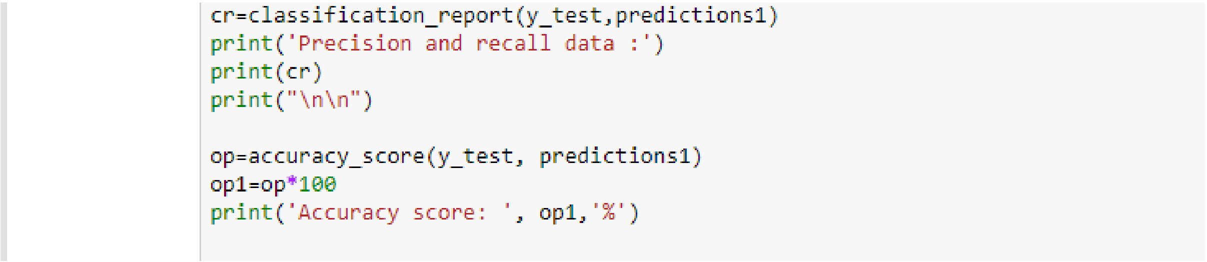
8.Random Forest Classifier

- **Train_split and test_split**



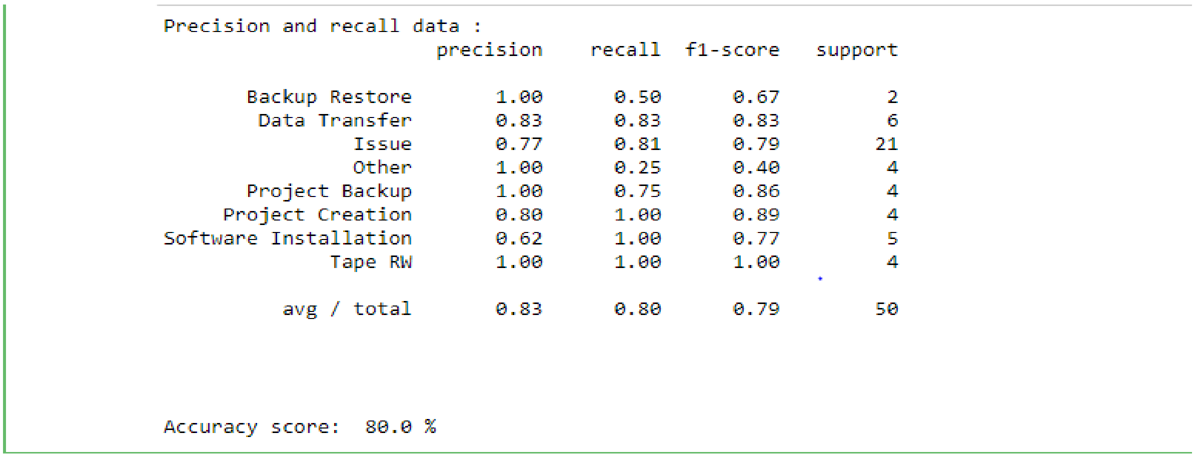
```
In [8]: from sklearn.ensemble import RandomForestClassifier

        rf = RandomForestClassifier(n_estimators = 200, random_state = 42)
        rf.fit(training_data, y_train)
        predictions1 = rf.predict(testing_data)
```

- **Model accuracy and precision**



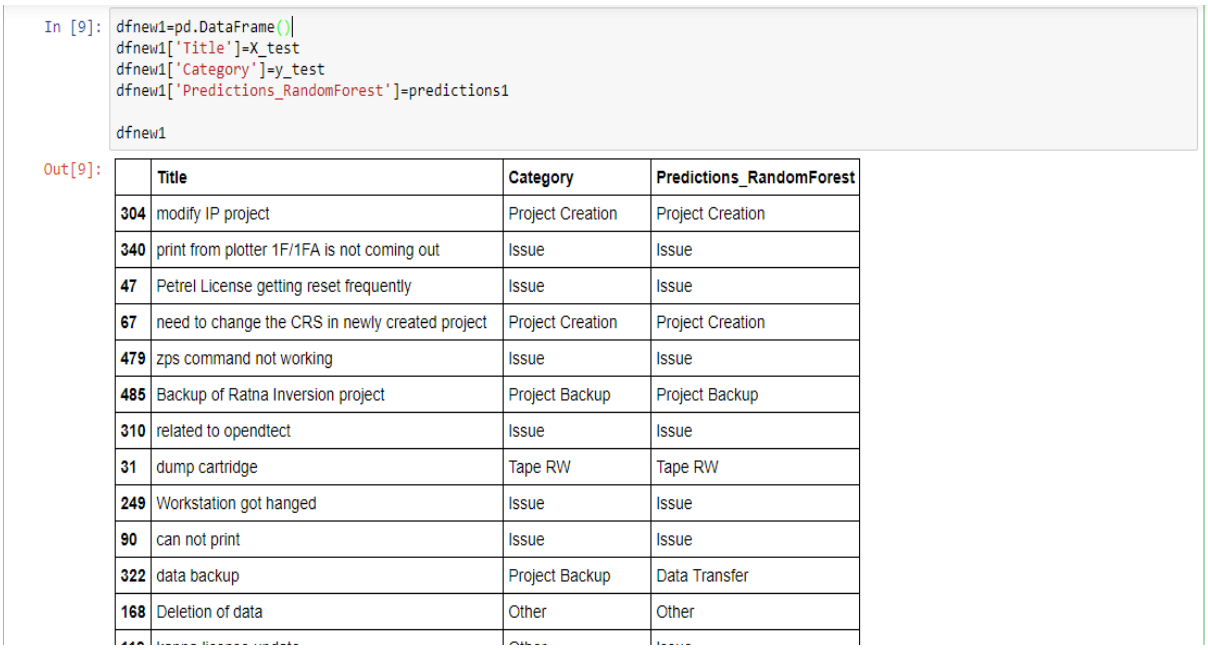
```
cr=classification_report(y_test,predictions1)
print('Precision and recall data :')
print(cr)
print("\n\n")

op=accuracy_score(y_test, predictions1)
op1=op*100
print('Accuracy score: ', op1,'%')
```

Output (step 8):



```
Precision and recall data :
                     precision    recall  f1-score   support

      Backup Restore       1.00      0.50      0.67         2
       Data Transfer       0.83      0.83      0.83         6
               Issue       0.77      0.81      0.79        21
               Other       1.00      0.25      0.40         4
      Project Backup       1.00      0.75      0.86         4
    Project Creation       0.80      1.00      0.89         4
Software Installation      0.62      1.00      0.77         5
             Tape RW       1.00      1.00      1.00         4

         avg / total       0.83      0.80      0.79        50



Accuracy score:  80.0 %
```

9.Viewing the result(RandomForest).

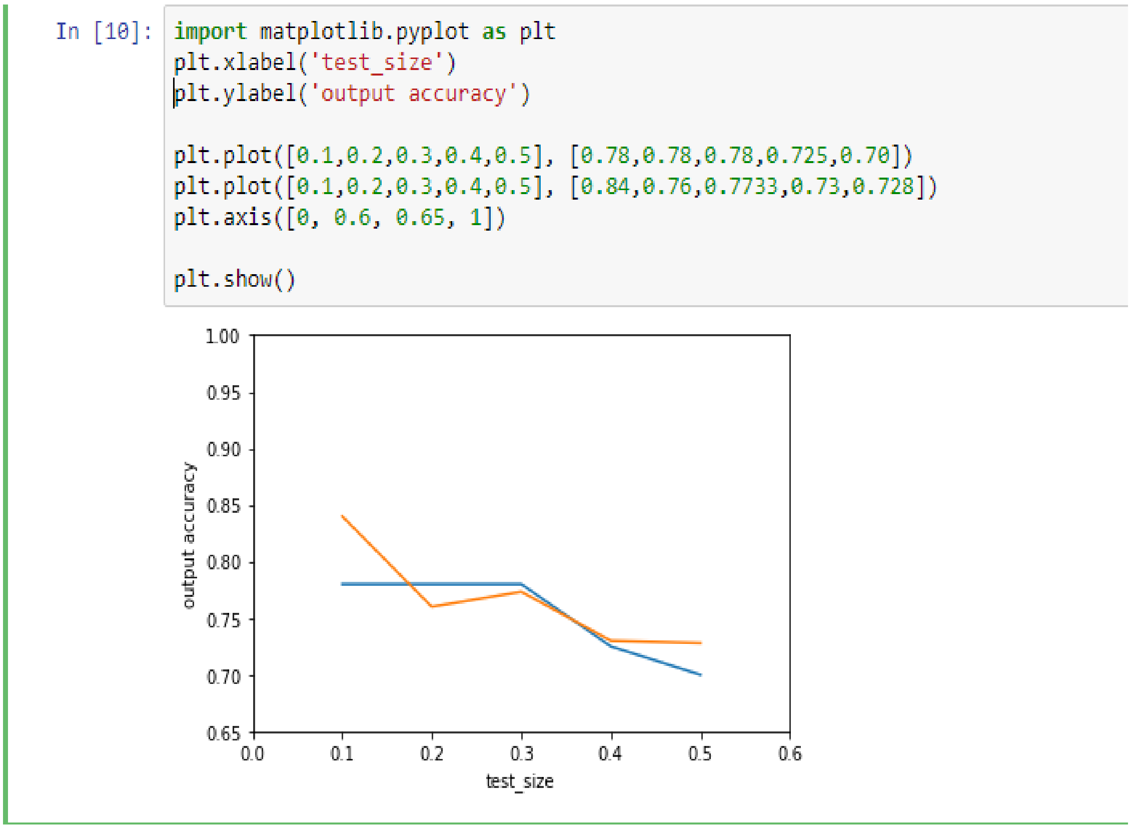


```
In [9]: dfnew1=pd.DataFrame()
        dfnew1['Title']=X_test
        dfnew1['Category']=y_test
        dfnew1['Predictions_RandomForest']=predictions1

        dfnew1
```

Out[9]:

	Title	Category	Predictions_RandomForest
304	modify IP project	Project Creation	Project Creation
340	print from plotter 1F/1FA is not coming out	Issue	Issue
47	Petrel License getting reset frequently	Issue	Issue
67	need to change the CRS in newly created project	Project Creation	Project Creation
479	zps command not working	Issue	Issue
485	Backup of Ratna Inversion project	Project Backup	Project Backup
310	related to opendtect	Issue	Issue
31	dump cartridge	Tape RW	Tape RW
249	Workstation got hanged	Issue	Issue
90	can not print	Issue	Issue
322	data backup	Project Backup	Data Transfer
168	Deletion of data	Other	Other
[illegible]	kappa license update	Other	Issue

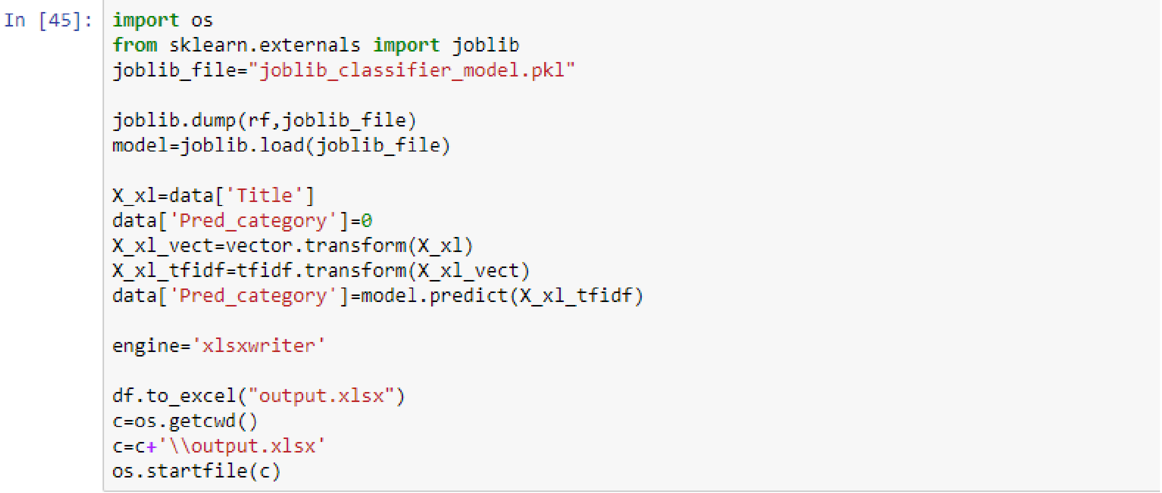
10.Plotting the accuracy graph



In [10]:

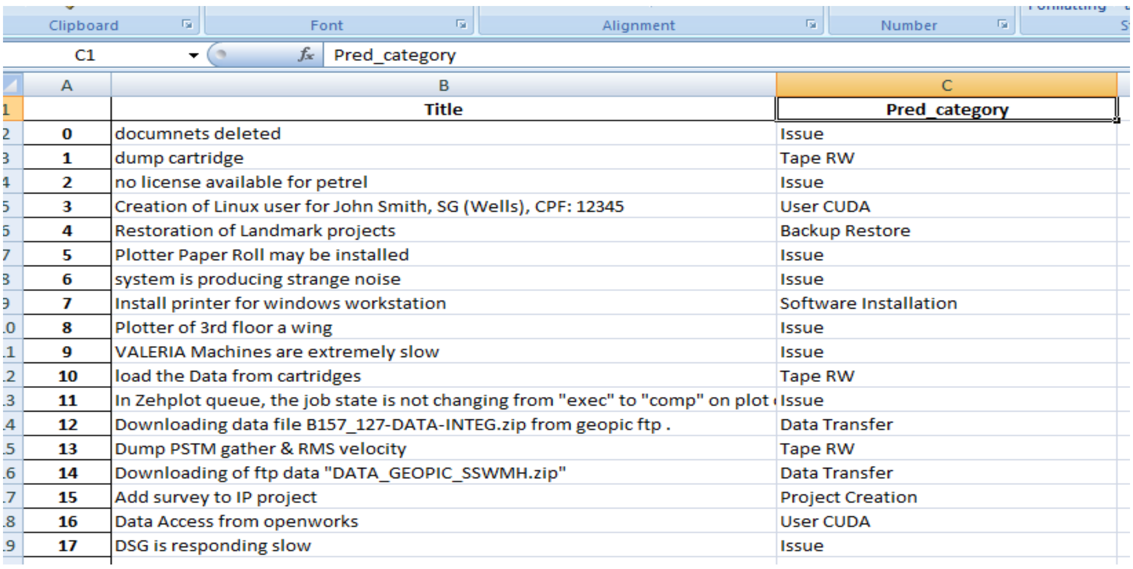
```
import matplotlib.pyplot as plt
plt.xlabel('test_size')
plt.ylabel('output accuracy')

plt.plot([0.1,0.2,0.3,0.4,0.5], [0.78,0.78,0.78,0.725,0.70])
plt.plot([0.1,0.2,0.3,0.4,0.5], [0.84,0.76,0.7733,0.73,0.728])
plt.axis([0, 0.6, 0.65, 1])

plt.show()
```

11.Saving and loading the file for future use



In [45]:

```
import os
from sklearn.externals import joblib
joblib_file="joblib_classifier_model.pkl"

joblib.dump(rf,joblib_file)
model=joblib.load(joblib_file)

X_xl=data['Title']
data['Pred_category']=0
X_xl_vect=vector.transform(X_xl)
X_xl_tfidf=tfidf.transform(X_xl_vect)
data['Pred_category']=model.predict(X_xl_tfidf)

engine='xlsxwriter'

df.to_excel("output.xlsx")
c=os.getcwd()
c=c+'\\output.xlsx'
os.startfile(c)
```



	A	B	C
1		Title	Pred_category
2	0	documnets deleted	Issue
3	1	dump cartridge	Tape RW
4	2	no license available for petrel	Issue
5	3	Creation of Linux user for John Smith, SG (Wells), CPF: 12345	User CUDA
5	4	Restoration of Landmark projects	Backup Restore
7	5	Plotter Paper Roll may be installed	Issue
3	6	system is producing strange noise	Issue
9	7	Install printer for windows workstation	Software Installation
.0	8	Plotter of 3rd floor a wing	Issue
.1	9	VALERIA Machines are extremely slow	Issue
.2	10	load the Data from cartridges	Tape RW
.3	11	In Zehplot queue, the job state is not changing from "exec" to "comp" on plot (	Issue
.4	12	Downloading data file B157_127-DATA-INTEG.zip from geopic ftp .	Data Transfer
.5	13	Dump PSTM gather & RMS velocity	Tape RW
.6	14	Downloading of ftp data "DATA_GEOPIC_SSWMH.zip"	Data Transfer
.7	15	Add survey to IP project	Project Creation
.8	16	Data Access from openworks	User CUDA
.9	17	DSG is responding slow	Issue

3.2 GUI

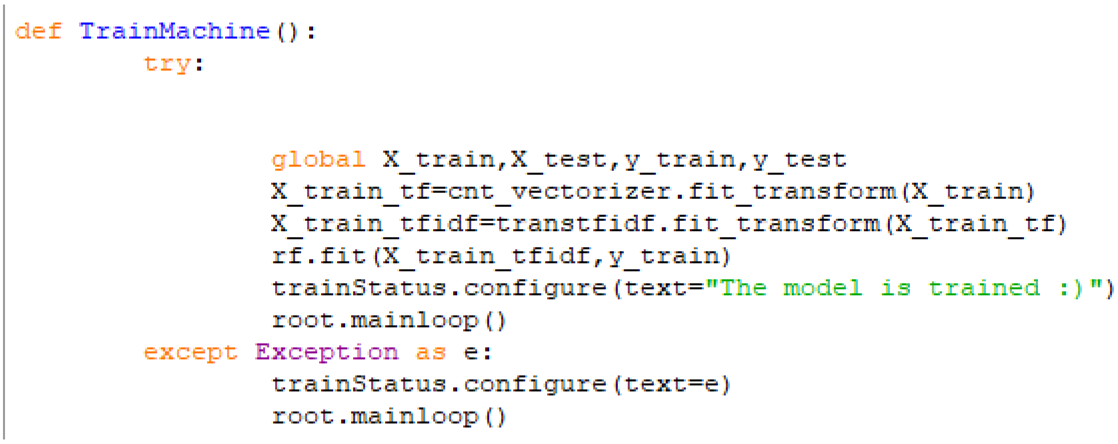
1.Select file

```
def FileSelect():
    try:
        global path
        StatusLabel.configure(text="Status:Stemming..Lematizing...Tossing..Turning..PLease Wait.")
        path=filedialog.askopenfilename(filetypes=(("Template files","*.xlsx"),("All files","*")))
        global df
        df=pd.read_excel(path)
        preprocessor(df['Title'],df)
        StatusLabel.configure(text="Status:File Path Configured...")
        df.Title.fillna(df.Title.dropna().max(),inplace =True)
        df.Category.fillna(df.Category.dropna().max(),inplace =True)
        root.mainloop()
    except Exception as e:
        StatusLabel.configure(text=e)
        root.mainloop()
```

2.Spliting the input data

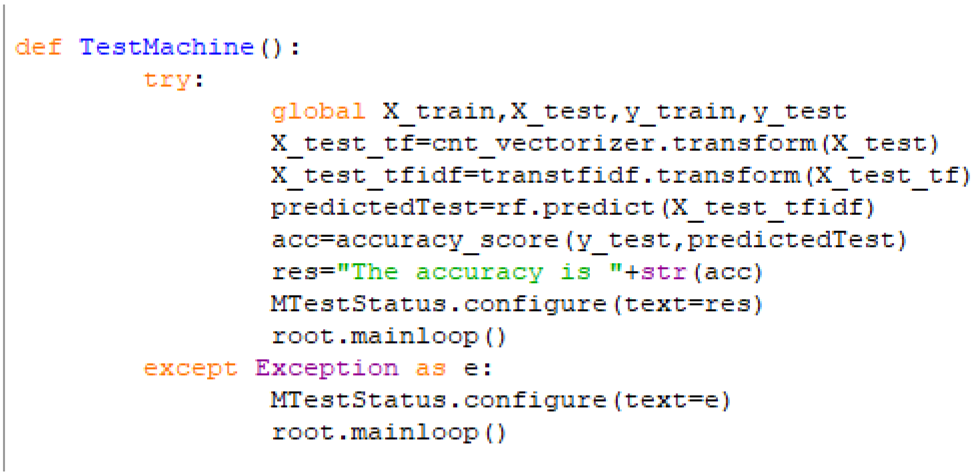
```
def SplitData():
    value=float(splt.get())
    try:
            global df
            global X_train,X_test,y_test,y_train
            X=df['Title']
            y=df['Category']
            X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=value,random_state=1)
            testStatus.configure(text="Splitted and Ready to workâ°")
            root.mainloop()
    except Exception as e:
            testStatus.configure(text=e)
            root.mainloop()
```

3.Train the model



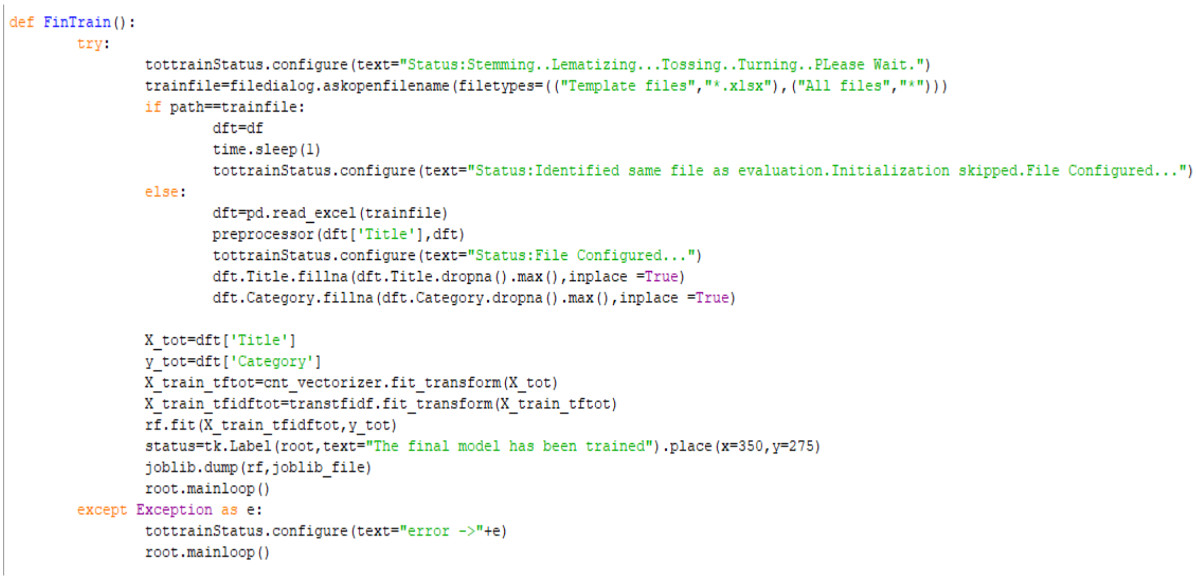
```
def TrainMachine():
        try:

                global X_train,X_test,y_train,y_test
                X_train_tf=cnt_vectorizer.fit_transform(X_train)
                X_train_tfidf=transtfidf.fit_transform(X_train_tf)
                rf.fit(X_train_tfidf,y_train)
                trainStatus.configure(text="The model is trained :)")
                root.mainloop()
        except Exception as e:
                trainStatus.configure(text=e)
                root.mainloop()
```

4. Test the model



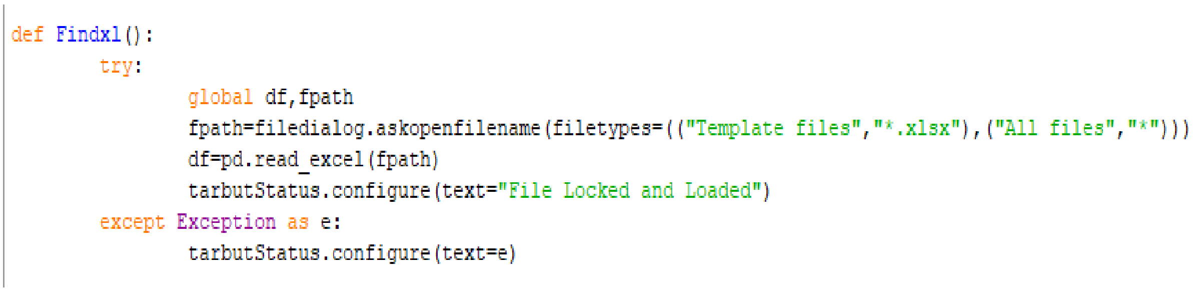
```
def TestMachine():
        try:
                global X_train,X_test,y_train,y_test
                X_test_tf=cnt_vectorizer.transform(X_test)
                X_test_tfidf=transtfidf.transform(X_test_tf)
                predictedTest=rf.predict(X_test_tfidf)
                acc=accuracy_score(y_test,predictedTest)
                res="The accuracy is "+str(acc)
                MTestStatus.configure(text=res)
                root.mainloop()
        except Exception as e:
                MTestStatus.configure(text=e)
                root.mainloop()
```

5. Final Training of machine



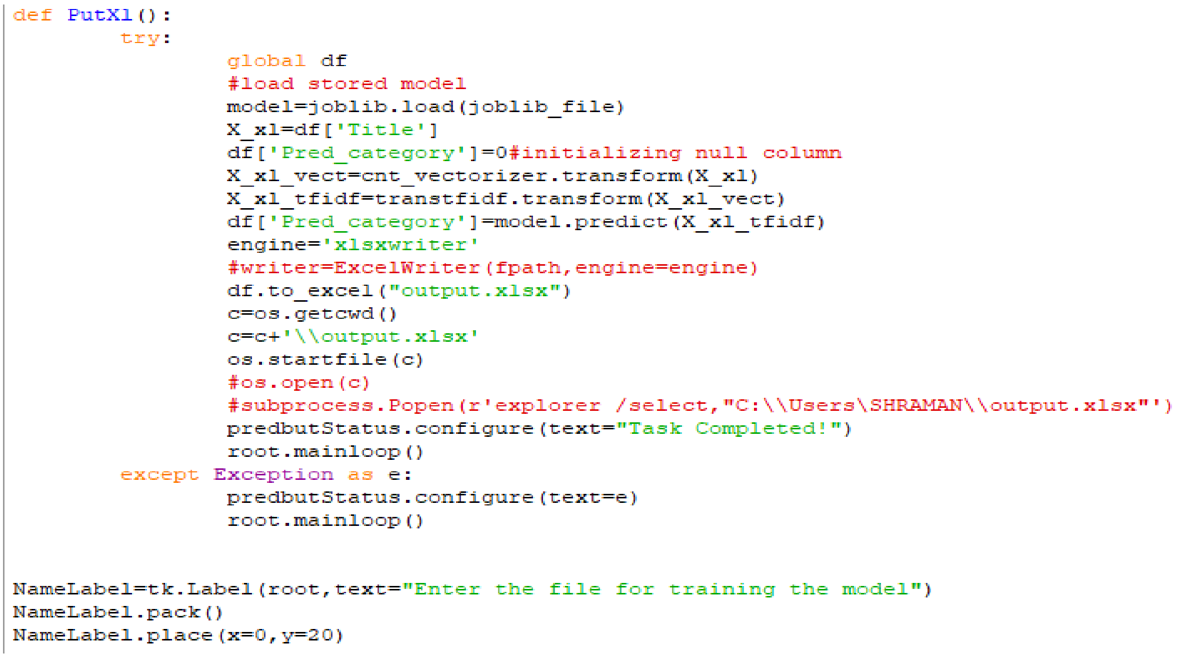
```
def FinTrain():
        try:
                tottrainStatus.configure(text="Status:Stemming..Lematizing...Tossing..Turning..PLease Wait.")
                trainfile=filedialog.askopenfilename(filetypes=(("Template files","*.xlsx"),("All files","*")))
                if path==trainfile:
                        dft=df
                        time.sleep(1)
                        tottrainStatus.configure(text="Status:Identified same file as evaluation.Initialization skipped.File Configured...")
                else:
                        dft=pd.read_excel(trainfile)
                        preprocessor(dft['Title'],dft)
                        tottrainStatus.configure(text="Status:File Configured...")
                        dft.Title.fillna(dft.Title.dropna().max(),inplace =True)
                        dft.Category.fillna(dft.Category.dropna().max(),inplace =True)

                X_tot=dft['Title']
                y_tot=dft['Category']
                X_train_tftot=cnt_vectorizer.fit_transform(X_tot)
                X_train_tfidftot=transtfidf.fit_transform(X_train_tftot)
                rf.fit(X_train_tfidftot,y_tot)
                status=tk.Label(root,text="The final model has been trained").place(x=350,y=275)
                joblib.dump(rf,joblib_file)
                root.mainloop()
        except Exception as e:
                tottrainStatus.configure(text="error ->"+e)
                root.mainloop()
```

6. Defining path



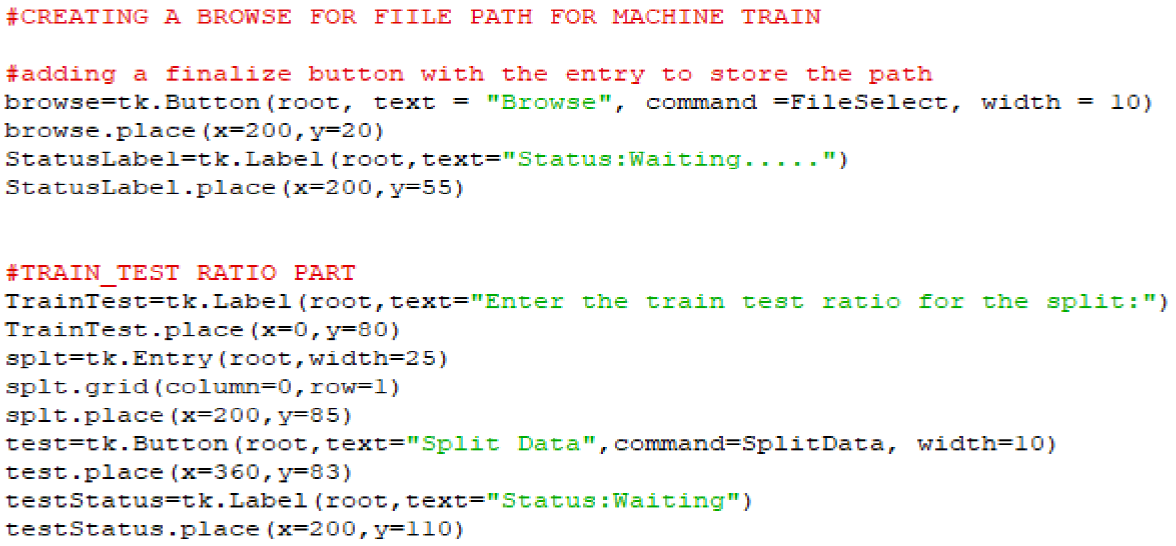
```
def Findxl():
        try:
                global df,fpath
                fpath=filedialog.askopenfilename(filetypes=(("Template files","*.xlsx"),("All files","*")))
                df=pd.read_excel(fpath)
                tarbutStatus.configure(text="File Locked and Loaded")
        except Exception as e:
                tarbutStatus.configure(text=e)
```

7.Storing the predicted category in Excel file



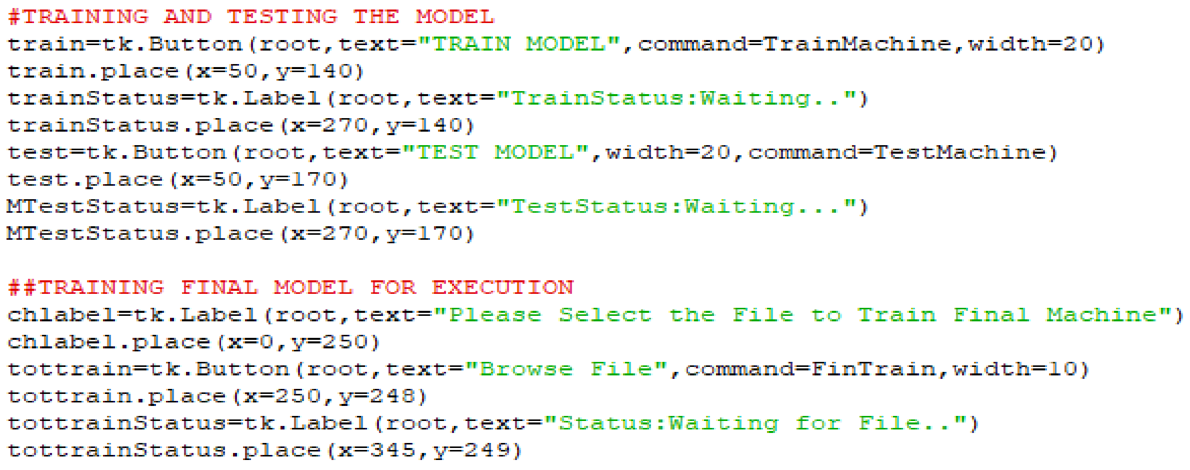
```
def PutXl():
        try:
                global df
                #load stored model
                model=joblib.load(joblib_file)
                X_xl=df['Title']
                df['Pred_category']=0#initializing null column
                X_xl_vect=cnt_vectorizer.transform(X_xl)
                X_xl_tfidf=transtfidf.transform(X_xl_vect)
                df['Pred_category']=model.predict(X_xl_tfidf)
                engine='xlsxwriter'
                #writer=ExcelWriter(fpath,engine=engine)
                df.to_excel("output.xlsx")
                c=os.getcwd()
                c=c+'\\output.xlsx'
                os.startfile(c)
                #os.open(c)
                #subprocess.Popen(r'explorer /select,"C:\\Users\SHRAMAN\\output.xlsx"')
                predbutStatus.configure(text="Task Completed!")
                root.mainloop()
        except Exception as e:
                predbutStatus.configure(text=e)
                root.mainloop()


NameLabel=tk.Label(root,text="Enter the file for training the model")
NameLabel.pack()
NameLabel.place(x=0,y=20)
```

8.Add ‘Browse’ button to interface



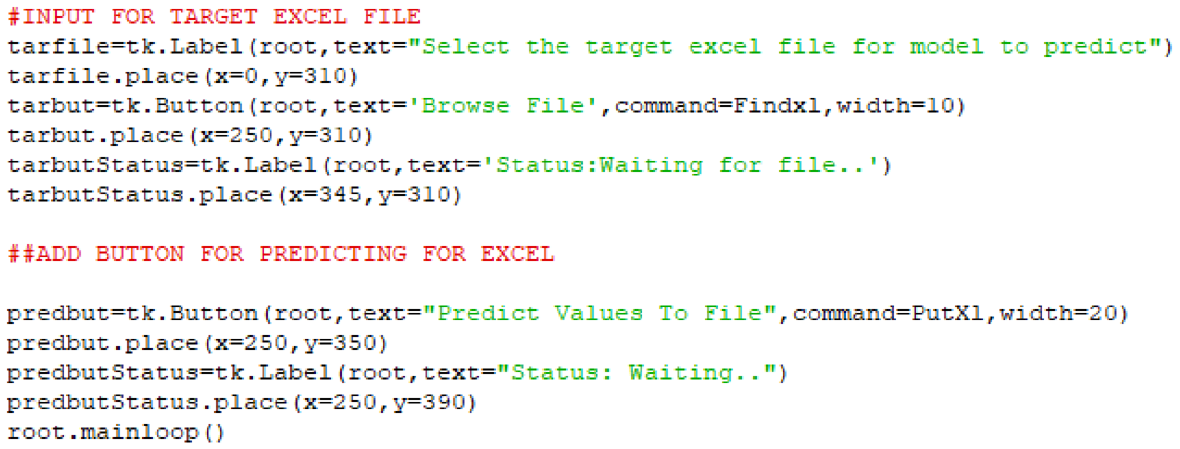
```
#CREATING A BROWSE FOR FIILE PATH FOR MACHINE TRAIN

#adding a finalize button with the entry to store the path
browse=tk.Button(root, text = "Browse", command =FileSelect, width = 10)
browse.place(x=200,y=20)
StatusLabel=tk.Label(root,text="Status:Waiting.....")
StatusLabel.place(x=200,y=55)


#TRAIN_TEST RATIO PART
TrainTest=tk.Label(root,text="Enter the train test ratio for the split:")
TrainTest.place(x=0,y=80)
splt=tk.Entry(root,width=25)
splt.grid(column=0,row=1)
splt.place(x=200,y=85)
test=tk.Button(root,text="Split Data",command=SplitData, width=10)
test.place(x=360,y=83)
testStatus=tk.Label(root,text="Status:Waiting")
testStatus.place(x=200,y=110)
```

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



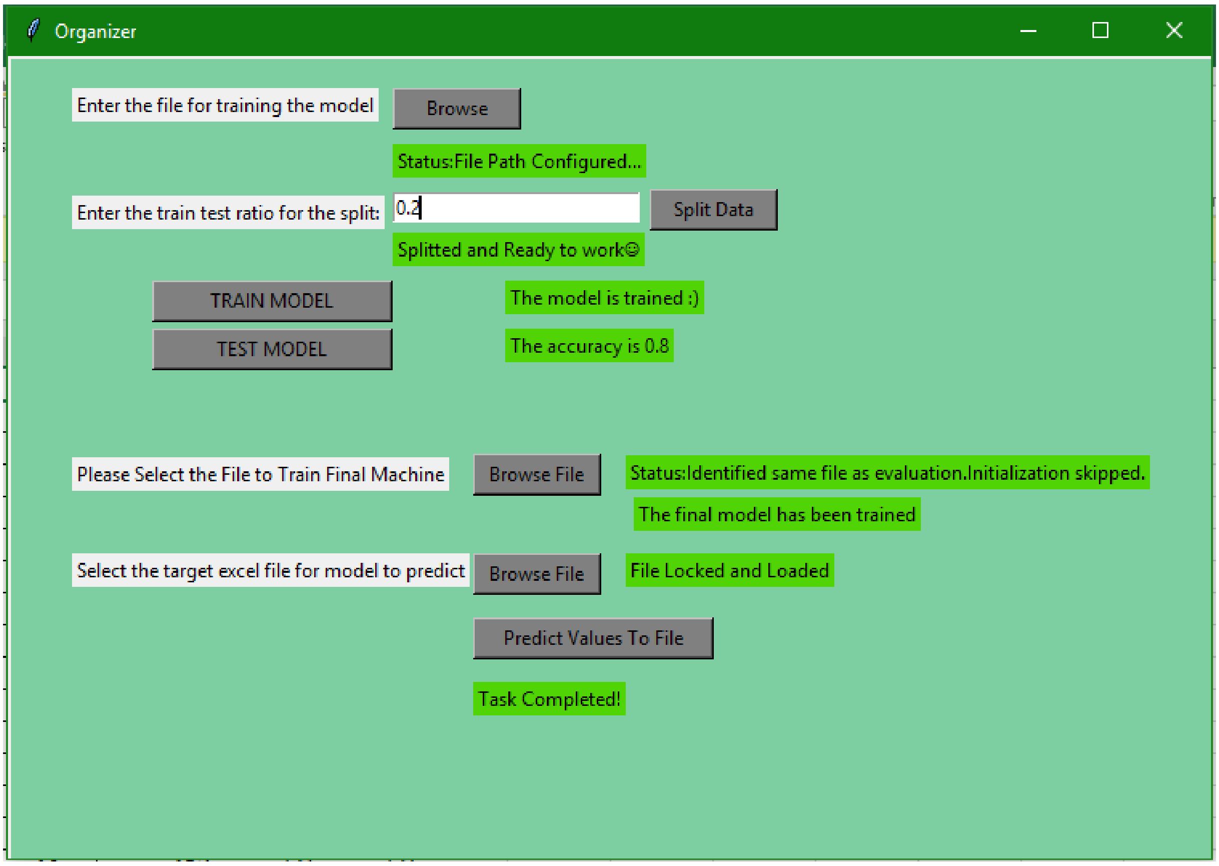
```
#TRAINING AND TESTING THE MODEL
train=tk.Button(root,text="TRAIN MODEL",command=TrainMachine,width=20)
train.place(x=50,y=140)
trainStatus=tk.Label(root,text="TrainStatus:Waiting..")
trainStatus.place(x=270,y=140)
test=tk.Button(root,text="TEST MODEL",width=20,command=TestMachine)
test.place(x=50,y=170)
MTestStatus=tk.Label(root,text="TestStatus:Waiting...")
MTestStatus.place(x=270,y=170)

##TRAINING FINAL MODEL FOR EXECUTION
chlabel=tk.Label(root,text="Please Select the File to Train Final Machine")
chlabel.place(x=0,y=250)
tottrain=tk.Button(root,text="Browse File",command=FinTrain,width=10)
tottrain.place(x=250,y=248)
tottrainStatus=tk.Label(root,text="Status:Waiting for File..")
tottrainStatus.place(x=345,y=249)
```

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



```
#INPUT FOR TARGET EXCEL FILE
tarfile=tk.Label(root,text="Select the target excel file for model to predict")
tarfile.place(x=0,y=310)
tarbut=tk.Button(root,text='Browse File',command=Findxl,width=10)
tarbut.place(x=250,y=310)
tarbutStatus=tk.Label(root,text='Status:Waiting for file..')
tarbutStatus.place(x=345,y=310)

##ADD BUTTON FOR PREDICTING FOR EXCEL

predbut=tk.Button(root,text="Predict Values To File",command=PutXl,width=20)
predbut.place(x=250,y=350)
predbutStatus=tk.Label(root,text="Status: Waiting..")
predbutStatus.place(x=250,y=390)
root.mainloop()
```

Output of GUI –



Enter the file for training the model

Status:File Path Configured...

Enter the train test ratio for the split: 0.2

Splitted and Ready to work☺

The model is trained :)

The accuracy is 0.8

Please Select the File to Train Final Machine

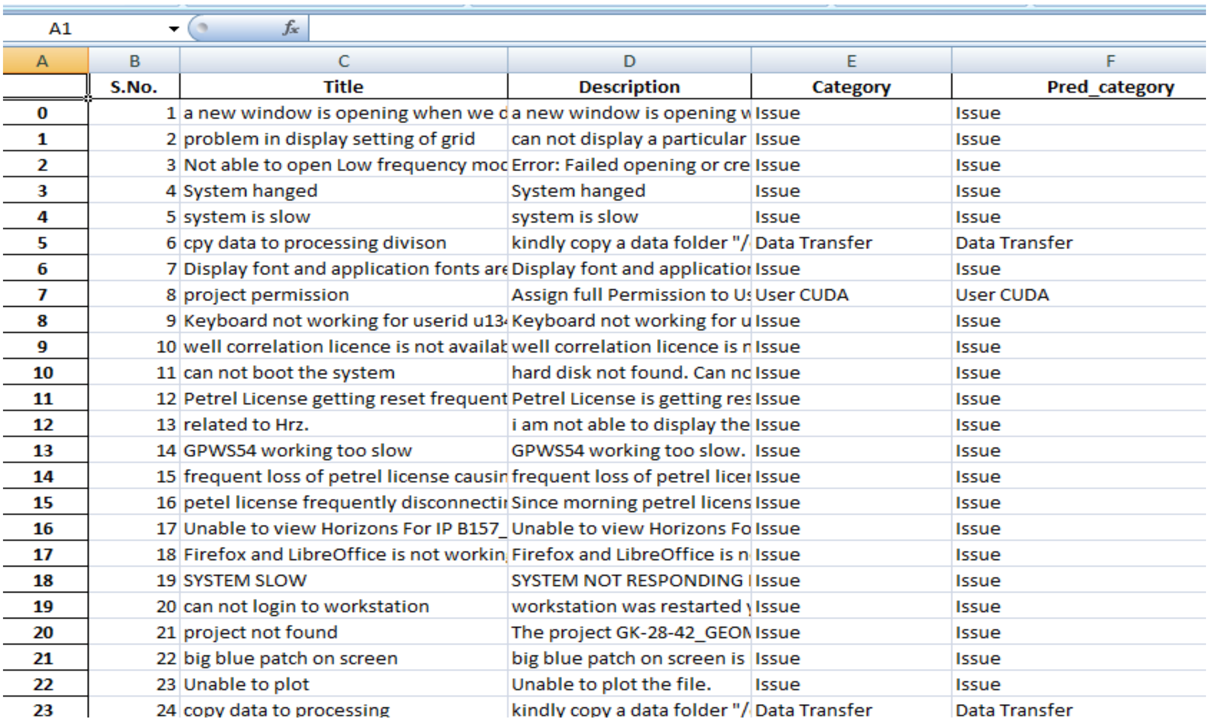
Status:Identified same file as evaluation.Initialization skipped.

The final model has been trained

Select the target excel file for model to predict

File Locked and Loaded

Task Completed!



	S.No.	Title	Description	Category	Pred_category
0	1	a new window is opening when we c	a new window is opening w	Issue	Issue
1	2	problem in display setting of grid	can not display a particular	Issue	Issue
2	3	Not able to open Low frequency moc	Error: Failed opening or cre	Issue	Issue
3	4	System hanged	System hanged	Issue	Issue
4	5	system is slow	system is slow	Issue	Issue
5	6	cpy data to processing divison	kindly copy a data folder "/	Data Transfer	Data Transfer
6	7	Display font and application fonts are	Display font and applicatio	Issue	Issue
7	8	project permission	Assign full Permission to U	User CUDA	User CUDA
8	9	Keyboard not working for userid u13	Keyboard not working for u	Issue	Issue
9	10	well correlation licence is not availat	well correlation licence is n	Issue	Issue
10	11	can not boot the system	hard disk not found. Can nc	Issue	Issue
11	12	Petrel License getting reset frequent	Petrel License is getting res	Issue	Issue
12	13	related to Hrz.	i am not able to display the	Issue	Issue
13	14	GPWS54 working too slow	GPWS54 working too slow.	Issue	Issue
14	15	frequent loss of petrel license causir	frequent loss of petrel licer	Issue	Issue
15	16	petel license frequently disconnectii	Since morning petrel licens	Issue	Issue
16	17	Unable to view Horizons For IP B157_	Unable to view Horizons Fo	Issue	Issue
17	18	Firefox and LibreOffice is not workin	Firefox and LibreOffice is n	Issue	Issue
18	19	SYSTEM SLOW	SYSTEM NOT RESPONDING	Issue	Issue
19	20	can not login to workstation	workstation was restarted	Issue	Issue
20	21	project not found	The project GK-28-42_GEON	Issue	Issue
21	22	big blue patch on screen	big blue patch on screen is	Issue	Issue
22	23	Unable to plot	Unable to plot the file.	Issue	Issue
23	24	copy data to processing	kindly copy a data folder "/	Data Transfer	Data Transfer

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 , 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, 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), 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/

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